SCALABLE ALGORITHMS AND DESIGN FOR DEBUG HARDWARE FOR TEST, VALIDATION AND SECURITY OF MIXED SIGNAL/RF CIRCUITS AND SYSTEMS

A Dissertation
Presented to
The Academic Faculty

by

Sabyasachi Deyati

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
Electrical & Computer Engineering

Georgia Institute of Technology
August 2017

SCALABLE ALGORITHMS AND DESIGN FOR DEBUG

HARDWARE FOR TEST, VALIDATION AND SECURITY OF

MIXED SIGNAL/RF CIRCUITS AND SYSTEMS

Approved by:

Dr. Abhijit Chatterjee, Advisor
School of Electrical & Computer Engineering
Georgia Institute of Technology

Dr. Arijit Raychowdhury
School of Electrical & Computer Engineering
Georgia Institute of Technology

Dr. Jennifer O Hasler
School of Electrical & Computer Engineering
Georgia Institute of Technology

Dr. Adit D. Singh
Department of Electrical & Computer Engineering
Auburn University

Dr. Saibal Mukhopadhyay
School of Electrical & Computer Engineering
Georgia Institute of Technology

Date Approved: June 2nd, 2017

ACKNOWLEDGEMENTS

I am indebted to many individuals who in one way or other have extended their support in my academic pursuit of accomplishing PhD. Firstly, with immense gratitude, I would like to acknowledge my PhD adviser Dr. Chatterjee for his relentless support and encouragement to do research. Without his insightful technical advice and moral support this dissertation would have not been possible. I also would like to thank Dr. Adit D. Singh for numerous intense discussions and his valuable feedback in my research. I would like to thank my other PhD committee members, Dr. Jennifer Hasler, Dr. Saibal Mukhopadhyay and Dr. Arijit Raychowdhury for reviewing my research and providing valuable feedback.

I would like to acknowledge the support of Intel Corporation, National Science Foundation and Semiconductor Research Corporation for funding my research at various stages of my PhD in Georgia Tech. I had the opportunity to work with many talented individuals during my internship at Intel Corporation. Especially I would like to mention Dr. Bruce Querbach and Dr. Nagib Hakim for their support and technical input in my research.

I also would like to thank my lab mate and friend Barry J. Muldrey for his unselfish collaboration and help during my stay at Georgia Tech. I consider it is an honor to work with my bright and talented colleagues and friends here in Georgia Tech. I wish to mention Dr. Debesh Bhatta, Dr. Debashis Banerjee, Dr. Xian Wang, Dr. Nicolas Tzou, Dr. Thomas Moon, Dr. Josua Wells, Suvadeep Banerjee, Md Imran Momtaz, Sujay Pandey for taking part in numerous thought provoking discussions during my PhD.

Lastly, I wish to mention my parents and sister for their unconditional love and encouraging me to pursue PhD. Without their moral support this journey would have not been possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

ADC	Analog to Digital Converter
ATE	Automated Test Equipment
BIST	Built in Self-Test
DAC	Digital to Analog Converter
DFT	Design for Test
DSP	Digital Signal Processing
EVM	Error Vector Magnitude
FFT	Fast Fourier Transform
GA	Genetic Algorithm
IC	Integrated Circuit
IIP3	Input Third Order Intercept
IMD	Intermodulation Distortion
LNA	Low Noise Amplifier
LO	Local Oscillator
MARS	Multivariate Adaptive Regression Splines
NF	Noise Figure
NN	Neural Network
PA	Power Amplifier
PNN	Probabilistic Neural Network
PUF	Physically Unclonable Function
RF	Radio Frequency
SNR	Signal to Noise Ratio

SUMMARY

With the advent of SOCs and SOPs, more functionalities are integrated into an IC or package. Higher level of integration has made testing, validation of ICs more challenging due to lack of observability of internal circuit nodes. This calls for new embedded Design for Test (DFT) circuit design and test methodology development. This work is geared towards solving the analog/RF testing problem mentioned above by crafting intelligent stimulus to excite the non-idealities of the circuit, along with machine learning algorithms to learn the behavior of the system. Though the manufacturing cost of a transistor is decreasing over the technology generations, test cost per transistor is remaining constant or decreasing at a lower rate. So, there will be a time when test cost per transistor will be more than the actual manufacturing cost of a transistor. Every newer technology advancement entails newer test methodologies for keeping the test cost at a certain bound. ATE cost for mixed-signal and RF ICs are higher than that of digital ICs. There is a need in industry for low cost efficient testing, tuning and validation methodologies for mixed-signal and RF circuits and systems. In this thesis, we have addressed the following validation problems:

i) Manufacturing testing (Process Adaptive RF Transceiver Testing)

ii) Post manufacture tuning (Learning Assisted Parallel Testing and Tuning of Massively Beam-forming MIMO systems)

iii) Pre and post silicon verification (Built In State Consistency Checking for Mixed-Signal/RF Verification)

We have found that the knowledge of DFx design for analog/RF circuits (stimulus design, sensor design) can be leveraged in other emerging security solutions also. For example, in Trojan detection in digital systems and in designing Physically Unclonable Functions (PUFs).

.

CHAPTER 1. INTRODUCTION

Aggressive scaling of technology nodes has enabled in accommodating more number of transistors within the same die area. More number of transistors within the same die area led to incorporate newer functionalities and features into an IC. With the advent of SOCs and SOPs, more functionalities are integrated into an IC or package. Higher level of integration has made testing, validation of ICs more challenging due to lack of observability of internal circuit nodes. For digital circuits, scan chain is a plausible solution to observe internal nodes. However, no such scan chain technique is available for analog, mixed-signal and RF circuits due to signal integrity and loading issues. Testing and validating integrated ICs are becoming challenging with newer technology nodes. This calls for new embedded Design for Test (DFT) circuit design and test methodology development. It is clearly pointed out in ITRS 2014 test document that multi-level packaging will impose new requirements for system level testing as access to individual dies are limited. And this entails significant Design for Test (DFT) improvements in highly integrated ICs. Alternative test and validation solutions such as Built in Self-Test (BIST), DFx are necessary for highly integrated systems. For BIST and DFx special analog signal generation (test stimulus) and analog signal capturing sensor capabilities are required.

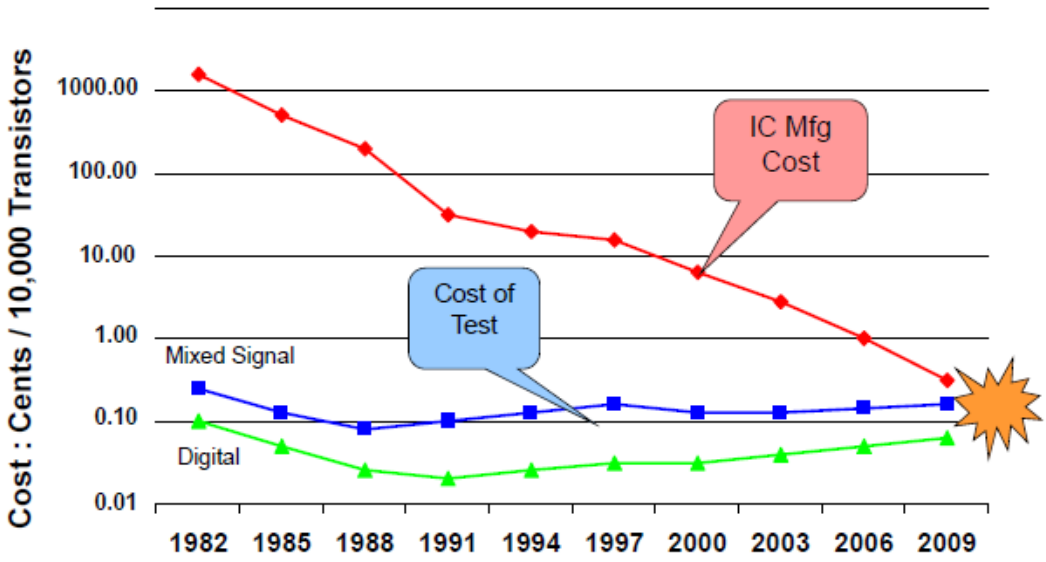


Figure 1: IC manufacturing and test cost

(Source: SOC Design Lecture 21 J A Abraham)

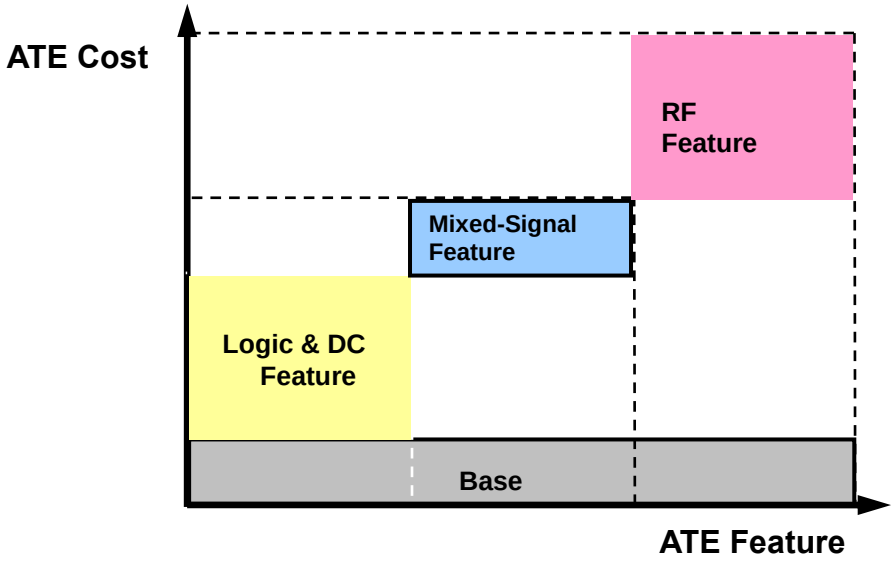


Figure 2: ATE cost for digital, mixed-signal and RF ICs

As shown in Figure 1, though the manufacturing cost of a transistor is decreasing over the technology generations, test cost per transistor is remaining constant or decreasing at a lower rate. So, there will be a time when test cost per transistor will be more than the actual manufacturing cost of a transistor. Every newer technology advancement entails newer test methodologies for keeping the test cost at a certain bound. As shown in Figure 2, ATE cost for mixed-signal and RF ICs are higher than that of digital ICs. There is a need in industry for low cost efficient testing, tuning and validation methodologies for mixed-signal and RF circuits and systems. In this thesis, we have addressed the following validation problems:

i) Manufacturing testing (Process Adaptive RF Transceiver Testing)

ii) Post manufacture tuning (Learning Assisted Parallel Testing and Tuning of Massively Beam-forming MIMO systems)

iii) Pre and post silicon verification (Built In State Consistency Checking for Mixed-Signal/RF Verification)

We have found that the knowledge of DFx design for analog/RF circuits (stimulus design, sensor design) can be leveraged in other emerging security solutions also. For example, in Trojan detection in digital systems and in designing Physically Unclonable Functions (PUFs). Instead of analog signal optimization we will use a pulse as a stimulus and several voltage and current sensors will be placed in the design to predict the presence of Trojans in digital pipeline systems. Challenge engineering (finding a suitable challenge for a PUF) is similar to stimulus optimizing problem of RF circuits.

The rest of the thesis is arranged as follows: In chapter 2, we have discussed process adaptive testing of RF transceiver[3, 4]. In chapter 3, we have discussed model based post silicon validation techniques using validation signatures [5-9]. In chapter 4, we have developed a model less methodology for pre and post silicon verification of mixed-signal and RF systems [10].In chapter 5 we have discussed parallel testing and tuning of massive beam-forming MIMO systems. In chapter 6, we have demonstrated how the state space representation of an analog circuit (described in detail in chapter 4, for analog verification) can help in designing and quantifying analog physically unclonable functions (PUFs) [11]. In chapter 7, we have developed a pulse based testing methodology for Trojan detection in digital circuits [12-14]. Pulse input is popular in analog, mixed-signal and RF testing domain.

CHAPTER 2. ADAPTIVE TESTING OF ANALOG/RF CIRCUITS USING HARDWARE-EXTRACTED FSM MODELS

2.1 Introduction

There been significant work in the past on the problem of signature-based alternative test of mixed-signal/RF circuits and systems. Such testing techniques are predicated on either: (a) alternative test stimulus generation algorithms that use circuit-level or behavioral models as core simulation engines [15-18], (b) careful test stimulus selection from test suites already constructed for the purpose of measuring device specification [19]and (c) application of random or pseudo-random test stimulus [20]. Method (a), above, is limited by the time-complexity of accurate simulation and by the limitations of behavioral simulation algorithms (accuracy, inability to model specific parameters such as bias currents, EVM etc.). These limitations are particularly acute because test generation algorithms [15-18] require repeated circuit/system simulation and test generation for even "small" circuits can run into days of compute time. In addition, any inaccuracies involved, place significant burden on back-end test response calibration algorithms [21] that are used to predict a DUT's performance specifications from its test response signature. In contrast, methods (b) and (c) suffer from significantly larger test complexity (time, number/length of tests) in comparison to (a) as well as the *inability to automatically modify/optimize test stimulus* to minimize test application time without compromising failure coverage. All the methods (a-c) suffer from one other key limitation: it is very difficult to design test stimulus in a simulation environment that takes all the *non-idealities of the tester instrumentation itself into account*. While models of probe, pin, cable

and tester electronics are not always accurate, inclusion of such models along with models of the DUT makes test generation even more difficult and simulation-intensive. To this effect, a methodology for generating tests directly from repeated chip measurements performed iteratively on a mixed-signal tester was proposed in [22] and avoids all device simulation. However, this approach is impractical as it involves engagement of a production test system over long periods of time for test generation and because *all* the hardware devices must be retested every time the test stimulus needs to be adapted to accommodate outlier devices or shifting process conditions (this requires cataloging and storage of hardware devices for retesting). Virtual probe technique [23] for performance prediction of spatially correlated ICs from same wafer can be leveraged here for model training.

Key contributions of this work are described as follows:

1. ***Finite State Machine Modeling of RF/Analog Circuits:*** *An envelope simulating model ABCD-RFH has been developed for quick envelop simulation of high frequency RF circuits. The model considers transient analog wave-shape and can be extracted from actual hardware stimulation or from SPICE level simulation of the DUT. As ABCD-RFH is a finite state machine model, response for an arbitrary input stimulus is obtained by traversing the state machine in accordance with the input stimulus. As state traversal for a reasonably big FSM is computationally inexpensive, the model simulates a RF system in orders of magnitude less time in comparison to state of the art RF simulators.*

2. ***Stimulus Generation:*** *Using the FSM models generated, ultra-fast stimulus generation for classification and specification prediction is proposed in this work.*

3. ***Adaptive Signature Test:*** *The test methodology developed in this work is adaptive to process corners. Classification, Outlier detection and specification prediction adapts to process shift.*

The remaining part of the chapter is arranged as follows: A brief literature survey of test stimulus generation and adaptive testing of analog/RF systems are presented in section 2.2. The adaptive testing approach is explained in section 2.3. FSM model generation technique (ABCD-RFH) is discussed in Section 2.5. Circuit and system level simulation results are shown in section 2.6. In Section 2.7, hardware IC measurement results corroborates efficacy of the adaptive test methodology proposed in this chapter. Conclusions are made in section 2.8.

2.2 Prior Work

2.2.1 Test Stimulus Generation

The earliest research on alternative test generation [24] attempted to replace standard specification based test procedures with fast transient tests and relied on the use of *linear sensitivities* of the performance specifications of the device under test (DUT) to process variations (around the nominal process parameter values), to guide the test search. The approach was based on the observation that there exists a minimum deviation of each of n_p relevant process parameters in the presence of worst case tolerance bounds on all

other process parameters, that violates at least one specification of the device under test (DUT) under given DUT specification test limits. The cost function for test generation was formulated in such a way as to detect such a minimum deviation for each of the n_p process parameters using a transient test, thereby *implicitly testing* for the DUT standard test specifications. The result of the test procedure was a "pass" vs. "fail" decision with very low misclassification rate. This work was followed by the work of Voorakaranam, et. al. [15, 16], in which changes ΔS in the performance specifications of the DUT were represented as a linear transformation of multi-parameter process perturbations ΔX via a sensitivity matrix A as given by $\Delta S = A\Delta X$. The objective was to design a set of alternative test measurements M to minimize the least squares norm of $||A - DB||$ as given in Eq. 1.

$$\Delta S = A\Delta X \quad \Delta M = B\Delta X \tag{1}$$
$$S: Specifications\ X: Process\ Parameters$$
$$M: Measured\ Signature$$
$$\Delta S = AB^{-1}\Delta S = D\Delta S\ where\ D = AB^{-1}$$

If the norm $||A - DB|| = 0$, it is seen that for arbitrary test limits on the performance parameters of the DUT, corresponding test limits on the alternate measurements M can always be set in such a way that *all multi-parameter process perturbations* that violate any performance specification of the DUT are *guaranteed to be detected* by the determined alternate test measurements and their corresponding test limits. A key limitation of this approach is that it relies on the *linear sensitivities* of the test specifications *S* and alternative measurements *M* to perturbations in the process parameters *X*. Hence, the goodness of the generated test waveform for discriminating "good" vs. "bad" devices close to the device test specification limits is not addressed appropriately and can suffer for devices with weak or strong nonlinear input-output characteristics. To address

this issue, the works of [16, 21, 25, 26] used nonlinear regression models to evaluate how accurately the test specifications of the device under test (DUT) could be predicted from the DUT response to each candidate test waveform. This is, however, computationally expensive as large numbers of device instances M in Figure 3 are necessary for constructing the regression models and validating the accuracy of the same in each iteration of the genetic optimization approach of Figure 3, resulting in hours of test generation time for even simple RF modules such as amplifiers and mixers. Finally, there has been research in the use of random test stimulus also. In [20], the use of random modulated bit streams for testing and diagnosis of RF systems has been explored. The method yields accurate RF test and diagnosis results but incurs longer test times than the use of carefully optimized tests .While we do not focus on diagnosis of RF systems in this research as in [20], the key objective of this work is in scalable and tractable test stimulus optimization for pass/fail classification and alternative test of generic analog/RF circuits.

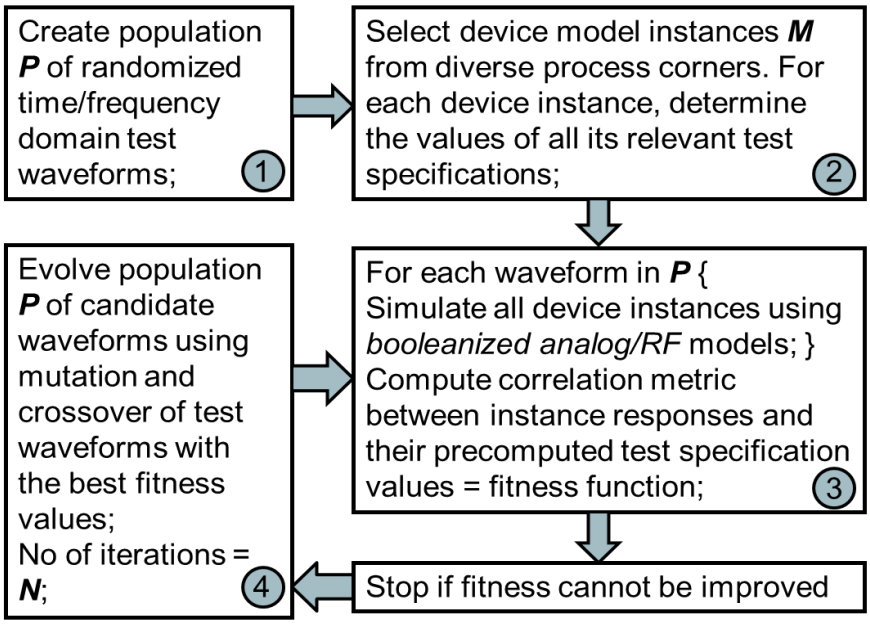


Figure 3: Test generation flow [16]

2.2.2 Adaptive Testing

Generally, the production tests are static in nature, i.e. the test patterns, characterizing models and test data analysis criteria for pass/fail do not change over time. If variation among the tested devices are small, static tests will serve the purpose of testing without much loss of accuracy. Process variation observed in today's RF/analog circuits manufactured in sub 22 nm technology node is huge as the gate oxide thickness and channel length are of the orders of diameters of few molecules. As alternate test is an indirect test based on statistical correlation of device specification and measured signature. Mapping of measured signature to device specification is a process dependent function. So alternate test based diagnosis, specification prediction calls for process adaptability.

There has been research on adaptive test ordering for early detection of fault and test time reduction[27] for AMS/RF circuits. An overview of adaptive testing for mixed signal circuits are given in [28]. In [29] the authors have demonstrated a per-device adaptive test for analog/RF circuits aiming at optimizing, compacting test sets. The authors have correctly pointed out that re-characterize the test methodology periodically at every wafer transition will involve unnecessary extra test time and cost. Dynamic process shift (wafer to wafer and lot to lot) is monitored by observing test statistics of few randomly selected devices subjected to full test suite (not compact test). These randomly selected devices are used to compute Kullback-Leibler distance from the characterization set. The proposed work of this chapter does not measure specification, predict specification from measured signature. So statistical distance measurement of specifications cannot be employed here. In [30] Stratigopoulos et al., have proposed adaptive alternate testing by employing a defect filter to screen out defective devices not suitable for alternate testing.

Wafer to wafer and lot to lot process adaptation were not incorporated in the test methodology.

2.3 Adaptive Testing Approach

Adaptive testing approach, proposed in this work is consisting of two parts: a) pre-silicon test development and b) post silicon adaptive manufacturing test development.

2.3.1 Pre-Silicon Test Development:

Steps in pre-silicon test development are shown in *Algorithm 1*. An ensemble of devices is created by uniformly sampling the multi-dimensional process parameter space. For every sampled device, a Booleanized model is extracted for fast transient simulation. Using the models of the sampled devices a stimulus is generated that can maximally separate the device responses. Each step in *Algorithm 1* is explained in detail below. (Booleanized model generation from SPICE simulation and from hardware stimulation is explained in section 2.5)

Algorithm 1: Pre-Silicon test development

1. *Create an ensemble of device models via uniform sampling of the multi-dimensional process parameter space of the manufacturing process*
2. *Create Booleanized models of each device in the ensemble using RF Booleanization algorithms*
3. *Design a test stimulus that allows minimal-size clustering of the devices in the ensemble based on the response of each device to the applied test*

2.3.1.1 Device Selection for Pre-Silicon Test

It is already mentioned in section II that test generation is a computationally expensive process. To reduce test generation time, we may not use all the available devices (hardware or software modeled). We will develop an algorithm to judiciously choose

devices such that the test quality is not compromised appreciably but test generation time is reduced.

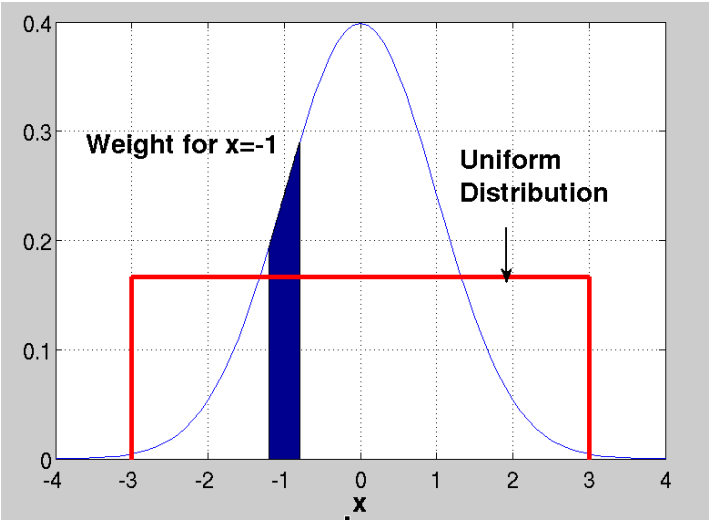


Figure 4: Probability weight for single variable

We assume that all the process parameters are normally distributed and the joint probability density function (pdf) of the process parameters is given by Eq. 2, below.

$$f(x_1, x_2 \ldots\ldots, x_n) = f(x) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp(-(x-\mu)^T \Sigma^{-1} (x-\mu)) \quad (2)$$

$where\ x_1, x_2, \ldots . x_n\ are\ n\ process\ parameters$
$\mu = [\mu_1\ \mu_2 \ldots \mu_n]\ mean\ of\ process\ parameters$
$\Sigma\ is\ covariance\ matrix\ of\ process\ parameters$

To create device instances corresponding to diverse process corners, a simple strategy is to generate devices by sampling the joint probability density function described by Eq. 2. However, this creates a large number of device instances centered around the mean of the joint pdf above. For test generation to be effective, we need instead many devices around the test specification acceptance boundaries of the DUT. This allows for effective discrimination of "good" vs. "bad" devices at the test specification limits allowing higher quality tests to be generated. To force this bias, we generate device instances using

a uniform distribution of the process parameters as shown in Figure 3, bounded by its 3σ limits. For every sampled device from the uniform sample space a weight is associated which is a measure of the probability of the device instance being generated if the process space was sampled from the joint normal distribution (Eq. 2). For a single process variable, this measure is the area under the normal curve around the sampled point (an example is shown in Figure 4 for x=-1). If the sampled device corresponds to the process parameters $(x_1', x_2', \ldots., x_n')$ the corresponding weight w for the corresponding device instance is given by Eq. 3.

$$w = \iiint_{\substack{x_1=x_1'-k*\sigma_1 \\ x_n=x_n'-k*\sigma_n}}^{\substack{x_1=x_1'+k*\sigma_1 \\ x_n=x_n'+k*\sigma_n}} f(x_1', x_2', \ldots., x_n') dx_1 dx_2 \ldots. dx_n \quad (3)$$

$$where\ k\ is\ a\ constant\ \ k = 0.1\ is\ chosen\ in\ this\ work$$

To ensure that device instances are selected from diverse process corners, a large number N of device instances are sampled as per the uniform process parameter distribution of Figure 3. Only a limited number M of N device instances are used. To select such M of N devices, first K random transient stimuli are generated. Subsequently, each of the N devices is stimulated by the K random test patterns. Consequently, every device instance is associated with K response vectors corresponding to time-sampled values of the response waveform. The distance d_{ij} between any two response vectors R_i and R_j corresponding to different devices *i* and *j* is given by the L2 norm of R_i and R_j(equation 4). The distance between two device instances *i* and *j*, D(i,j) is defined to be the largest distance across all the K random test stimuli (Eq. 5). To identify the M devices out of N, K-means

clustering is performed to cluster the N devices into M clusters in such a way that the mean distance between all devices in a cluster is minimized.

$$d_{ij} = \frac{1}{N}\sum_{t=1}^{N}\left(R_{it} - R_{jt}\right)^2 \quad (4)$$

$$D(i,j) = \max\ {d_{ij}}^k \quad (5)$$

$$where\ d_{ij}^k\ is\ the\ distance\ between\ response\ vectors\ \ R_i and\ R_j\ for\ k^{th}\ stimulus$$

The last step of the procedure consists of picking one device instance from each cluster to generate the M devices needed in box 2 of Figure 3. Let L be the vector of test specification limits for each of the test specifications of the DUT and let L_u be the corresponding vector of test specification values for the *u'th* device instance. We determine the device u in each cluster with the smallest norm $||L-L_u||$ (this is the device that is closest to the test acceptance limits of the DUT). When two or more devices have similar values of $||L-L_u||$, the device with the higher weight w, defined by Eq. 3 is selected from the cluster of device instances. Finding the multidimensional integral (Eq. 3) is not straight forward. For sake of brevity we are not discussing here how to enumerate this integral. The enthusiastic readers are requested to consult [31] for numerical integration techniques to compute this integral and [32] for computing the integral using mahalanobis distance. In this work the authors have used mahalanobis distance technique discussed in [32].

Algorithm 2: Device selection algorithm

1. N: Number of randomly generated devices;
2. M: Number of devices to be selected in box 2 of Figure 3.
3. Generate K random transient test stimuli;
4. Simulate all N device instances and capture response signatures corresponding to all K stimuli for each device;
5. For all pairs of devices i, j, compute the distance D(i,j);
6. Use k-means clustering algorithm to partition the devices into M clusters:
   Pick one device from each cluster based on its distance from the test specification limits, resolving conflicts via its weight "w" as per equation 3.

2.3.1.2 Pre-Silicon stimulus generation:

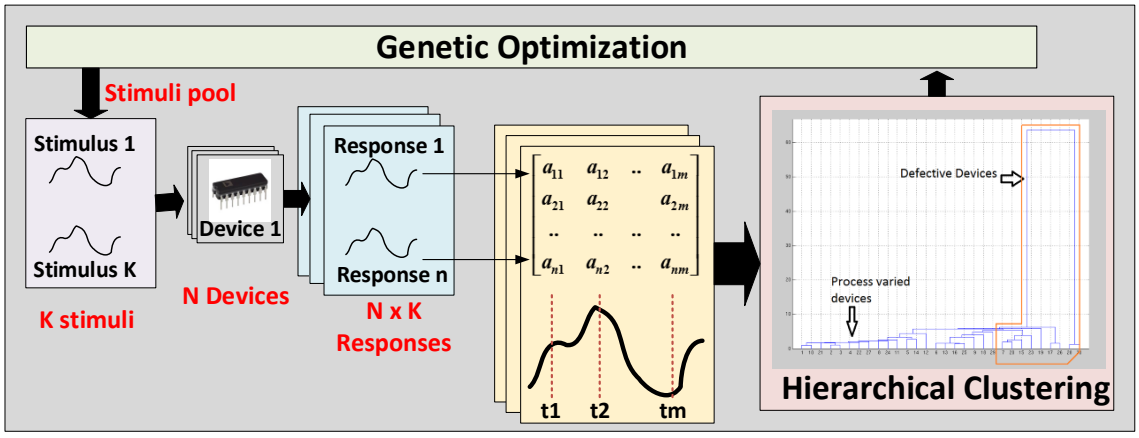


Figure 5: ATPG at pre-silicon stage

To discriminate amongst devices from diverse process corners, a hierarchical clustering algorithm is used in this research. The objective is to maximize the number of clusters (devices with similar behavior) for a given distance cutoff (a proxy measure of process variability). As the number of clusters is not known a-priori, this must evolve from simulation (IC measurement in post silicon) data. Let x_1 and x_2 be two input stimuli and NC_1 and NC_2 be the number of clusters corresponding to x_1 and x_2. If $NC_1 > NC_2$ then x_1 is preferred over x_2 as an input test stimulus since it classifies the process space into

larger numbers of clusters. Pre-silicon stimulus generation methodology is shown in Figure 5.K number of random stimuli are generated and applied on N number of sampled devices. Here every stimulus is a piecewise linear (PWL) wave. For every stimulus, we get N responses from N sampled devices. Total NxK responses are captured. For every stimulus, we run a hierarchical clustering on N responses and based on a predefined cutoff distance number of unique clusters is determined. A brief tutorial on hierarchical clustering is given in next section. For the same cutoff distance, the stimulus that can produce maximum number of unique clusters is better than all other stimuli as it is most sensitive among all the stimuli. The above description explains only one iteration of stimuli generation. The stimuli generation process goes on for multiple iteration with the help of genetic optimization (as shown in Figure 5). New pool of stimuli is generated from previous pool using genetic cross over and mutation. Pre-silicon stimuli is generated from process varied devices in simulation. It is used as a seed stimuli in post silicon stimuli generation and modified later in post-silicon stage. A pseudo code of pre-silicon stimuli generation is shown in Algorithm 3.

Algorithm 3: Pseudo code for pre-silicon stimulus generation

```
1.  Given: Circuit Netlists /Circuit Models / Circuit Hardware Instances from various process corners
2.  Objective: classify the given sample into maximum number of reasonable clusters where each cluster (class) will signify a varied process corner.
3.  Input = random input set
4.  While (stopping criteria not meet)
5.      For i=1 to Number of inputs in input set
6.          Signature=simulate (netlist, input(i))
7.      // Algorithm 4
8.      L(i)=hierarchical clustering (signature)
9.      Input = cross over (input ,L); // genetic cross over
10.     Input = mutate (input ,L);     // genetic mutation
```

11. Optimized stimulus = input(1)

2.3.1.3 Hierarchical clustering tutorial:

Hierarchical clustering is of two types a) agglomerative clustering (bottom up) and b) conglomerative clustering (top up). In this work, we have used bottom up clustering and a brief description of the hierarchical agglomerative clustering (HAC) is given below. HCA provides a hierarchy of clusters. Graphical representation of HCA is known as dendrogram. Each individual element to be clustered is considered a cluster at the beginning. Based on element features, pair wise distances are enumerated. Popular distance metrics are Euclidian, mahalanobis etc. After that in each iteration two minimally separated clusters are merged into one cluster and the process is continued till all the clusters are merged into one cluster. The above process creates a hierarchy of clusters. This hierarchy of clusters is graphically represented by a dendrogram. Based on user given cut off distance the dendrogram is cut (as shown in Figure 6) to generate the required clusters. In Figure 6, two examples are shown for cut off distance 1.1 (3 clusters) and 2 (2 clusters). A pseudo code for HAC is given in Algorithm 4.

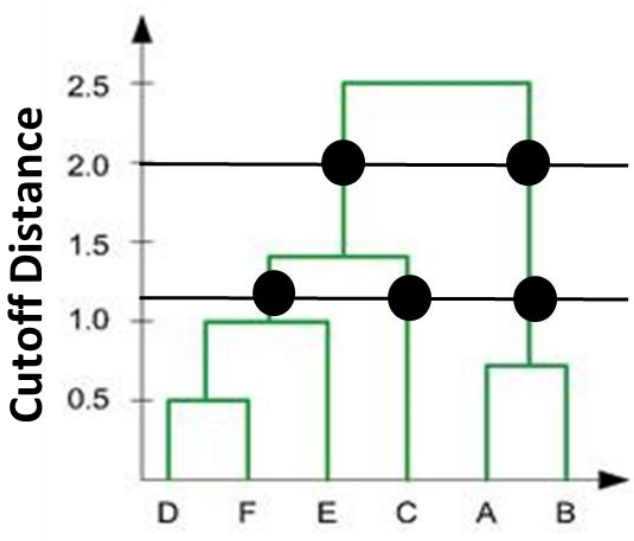


Figure 6: Dendrogram

Algorithm 4: Hierarchical agglomerative clustering (HAC)

1. For a given stimulus capture device responses (signatures).
2. Generate pair wise distance vectors for all captured responses in the previous step. (Popular distance metrics are Euclidian, Mahalanobis etc.)
3. Generate linkage tree (Figure 6) [33] based on distances calculated in the previous step. The linkage function (see Figure 6) links pairs of objects that are close together into binary clusters.
4. Based on user given cutoff distance, clusters are formed from the linkage tree constructed in step 3.

2.3.2 Post Silicon Adaptive Manufacturing Test Development:

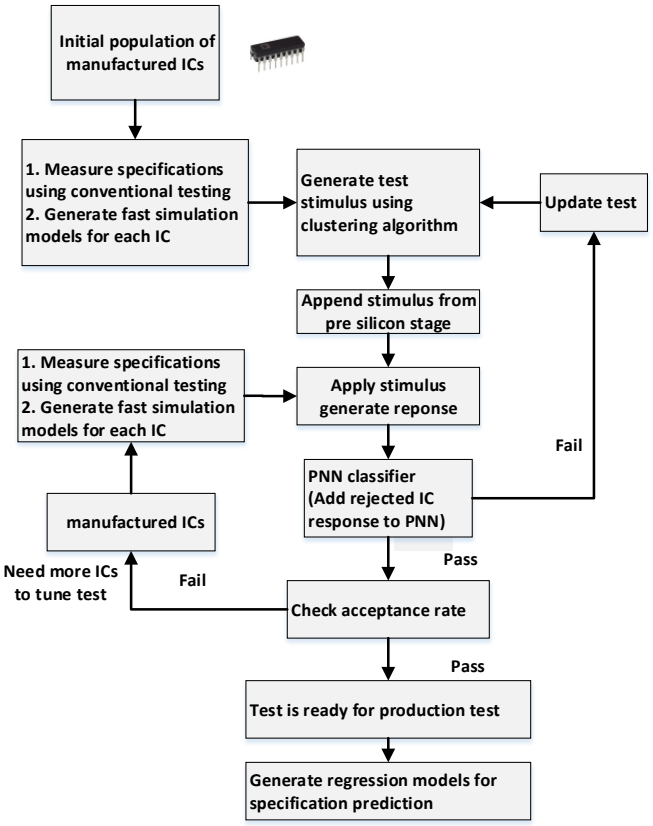


Figure 7: Post silicon adaptive test development flow

Post silicon adaptive test development flow (as shown in Figure 7) has three steps: i) Initial test development ii) Test tuning until it is ready for production level testing and iii) Generate regression models for specification prediction.

2.3.2.1 Initial Test Development:

Post-silicon initial test development is similar to pre-silicon test development. Instead of process varied netlists, manufactured ICs are used to generate the test from initial population of ICs.

2.3.2.2 Test Tuning:

N (N=200 is used in this research) number of ICs were used to generate the stimulus in the previous step. These N ICs are fully characterized i.e. their specification values were measured by conventional ways. As shown in Figure 7, stimulus generated in pre-silicon stage is appended to the stimulus generated in initial post silicon test development and the combined stimulus is used in the following stages for stimulation (response generation). These initial N ICs are used to create a Probabilistic Neural Network (PNN) classifier. Each IC response becomes a neuron in the PNN. Corresponding neuron classes are assigned based on the IC specification values. In this research, we have categorized ICs into three classes: good, marginal and bad respectively based on their measured specification values. A short tutorial on PNN is given in the next section. Now a newer set of N ICs (not used in building PNN) are taken and stimulated using the stimulus generated before. Using their responses, they are classified using the already built PNN. We use a threshold probability to accept a classification done by PNN. Let's assume the PNN is built from three types of devices: class 1, class 2 and class 3. When a new device response comes for classification, PNN finds the probabilities (p1 p2 and p3) that the device belongs to class 1, class 2 and class 3. For sake of argument if enumerated probabilities are 0.34, 0.33 and 0.33, PNN will indicate that the device belongs to class 1. As it is clearly visible from the numerical probability values that the margin or confidence of classification is poor, we would not accept this classification and will reject the device. At the tuning phase, a rejected device does not indicate that the device is functionally bad, it only indicates that the PNN has not got a similar device in its training set. Let's assume we set threshold

probability at 0.8 then any new device will be rejected, if the probability of the predicted class is below 0.8.

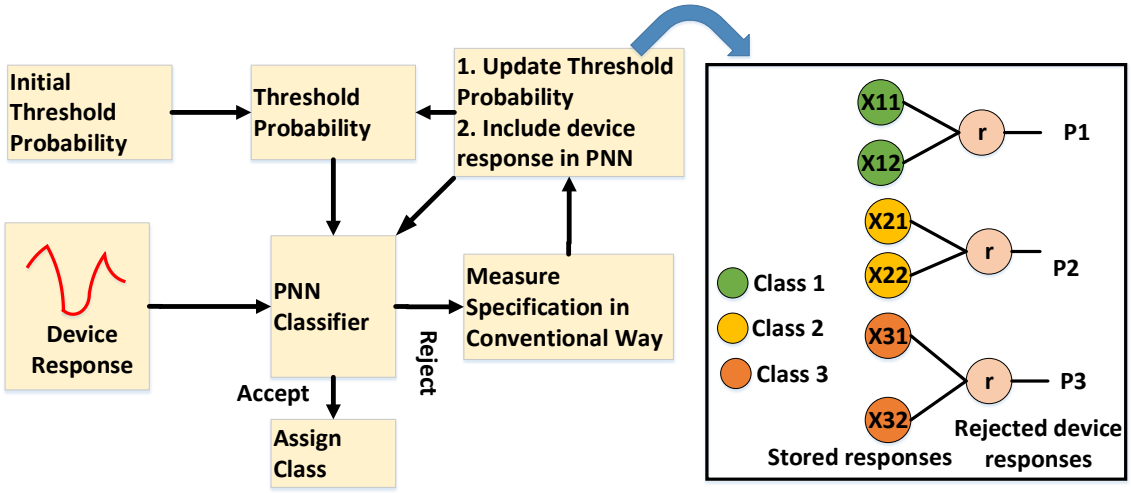


Figure 8: Threshold probability update procedure

In the model tuning stage, initially we start with a high threshold probability value and the threshold is reduced when new devices are added to the PNN. Threshold update procedure is shown in Figure 8. When a new device is rejected based on its signature response, we measure specifications of that device conventionally and from those measured specification values classify the device into one of the classes. Now we find the probability the device belongs to that class from the already stored device responses in the PNN (this probability is termed as P_{added}). New threshold probability is set as per Eq. 6 .

$$P_{new} = \frac{N * P_{old} + P_{added}}{N + 1} \tag{6}$$

P_{new}: $New\ Threshold\ Probability$ P_{old}: $Old\ Threshold\ Probability$
P_{added}: $Maximum\ class\ probability\ of\ newly\ added\ device$
N: $Number\ of\ neurons\ in\ the\ PNN$

The above-mentioned process of adding new devices to the PNN is continued till we reach a stage when the acceptance rate of the PNN is suitable for production testing. In this work, we have considered 99.9% acceptance rate as the bar for production testing. As shown in Figure 7, stimulus is also regenerated by adding all the initially failing devices into the stimulus generation population of devices. Adding a device response to the PNN for classification purpose is easy (shown in detail in PNN section), on the other hand stimulus generation is computationally expensive. So, as soon as a device fails PNN classification test, we add the device response to the PNN but wait for N devices to accumulate before regenerating the stimulus.

2.3.2.3 Generate Specification Prediction Test (SPT):

As shown in Figure 7, the last step in post-silicon test generation is regression test. Classification test only classifies devices into different classes(bins), but does not predict specification values. If devices are needed to be shipped with specification values, then this test is required. The objective of the SPT is to *maximize the correlation* between the test specification values of a sample of DUTs and the response of the DUTS to the applied SPT while *minimizing the sensitivity of the generated test stimulus to measurement noise*. Regression test is generated separately for all the classes. Test generation procedure is described below:

To compute the SPT, a candidate test stimulus is applied to *all the FSM models of DUTs of a class (devices are already classified by previous step)*. These include models for DUTs that may have been shipped to a customer. As per Figure 1, the standard

specification values of each of the DUTs are also known. Let the regression equation mapping signature to specification be given by Eq. 7.

$$Mx=c \tag{7}$$

M: signal Matrix
x: regression coefficients
c: measured specification values (gain, IP3 etc.)

Linear regression solution of x is given by Eq. 8 and Eq. 9.

$$x = (M^T M)^{-1} M^T c \tag{8}$$

$$let\ \ M^T c = b\ and\ M^T M = A\ so\ x = A^{-1} b \tag{9}$$

Where A is a real valued symmetric square matrix

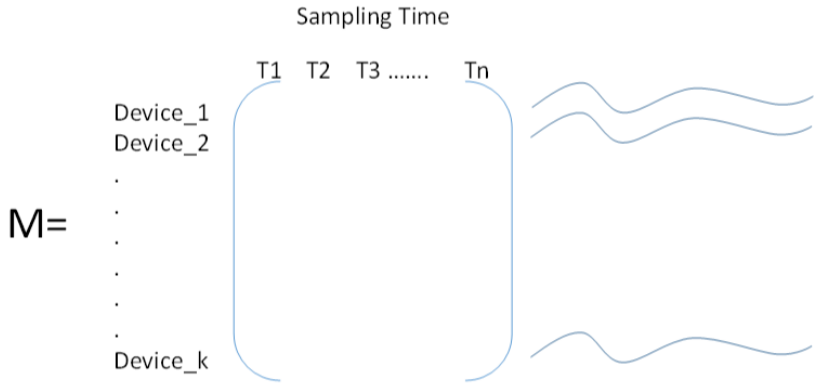


Figure 9: Signature matrix

In presence of noise (measurement noise, circuit noise) above Eq. 7 becomes Eq. 10.

$$(A + \delta A)(x + \delta x) = (b + \delta b) \tag{10}$$

$$\frac{||\delta x||}{||x||} \leq ||A^{-1}||(||A|| \frac{||\delta b||}{||b||} + \frac{||\ \delta A(x+\delta x)||}{||x||}) \tag{11}$$

From the above Eq. 11, we see that the error will be minimum when norm of A^{-1} and norm of A are minimum. In our test generation, the conditional number of A for an acceptable solution should be less than the user given error tolerance.

$$K(A) = \left||A|\right| * \left||A^{-1}|\right| < \sigma \tag{12}$$

If two variables are linearly dependent, then only Pearson's product moment correlation is applicable. If X is not a monotonic function of Y and if there is noise in measurement, then rank correlation is ineffective. Both the above-mentioned scenarios are prevalent in alternate transient signal measurement of an analog/RF IC. There is inherent noise in analog signal capturing and the relationship between measurement and specifications are highly nonlinear and their monotonicity cannot be guaranteed. BD (Brownian Distance) correlation [34]is a perfect dependency checking between alternate measurements and specifications. BD correlation is a measure of statistical dependence between two vectors of arbitrary length. If $(X_i, Y_i)\ \ i = 1 \ldots . n$ are samples of two vectors (X, Y) then distance correlation of the two vectors is given by Eq. 13.

$$dcor(X, Y) = \frac{dcov(X, Y)}{\sqrt{dvar(X) * dvar(Y)}} \tag{13}$$

Detailed derivation of Eq. 13 can be found in [34].Distance correlation is easy to enumerate compare to other proxy's for dependency checking used in signature testing previously. A runtime comparison among various proxies used for dependency checking is shown in Table 1.

Table 1: Average runtime comparison among various mappers used in alternate test stimulus generation

Feed Forward Neural Net	Linear Regression	MARS	Distance Correlation
60 secs	20 secs	100 secs	2 secs

Test generation proceeds by finding a suitable stimulus which maximizes the correlation between the DUT specifications and the response of each DUT to the applied test. This is formulated as given in Eq. 14.

$$\min \; c_1 + c_2 \ldots\ldots + c_k \qquad s.t \; K(A) < \sigma \tag{14}$$

c_i : Distance correlation [34] between specification i and alternate measurements

K (A): Conditional number of information matrix ($A = M^T M$)

σ : A user given value

The objective function of the minimization problem aims to pick that stimulus which will ensure that every specification is correlated to the alternate measurement. Any transient analog measurement is subjected to inevitable noise and measurement inaccuracy. If the conditional number of the information matrix formed from the alternate measurement is low, then the effect of noise and measurement inaccuracy can be avoided in alternate testing. Speed up in test generation is shown in Table 2.

Table 2: Speed up in test generation

Ref [21]	This work
6250 secs	11 secs

2.4 Probabilistic Neural Network Classifier

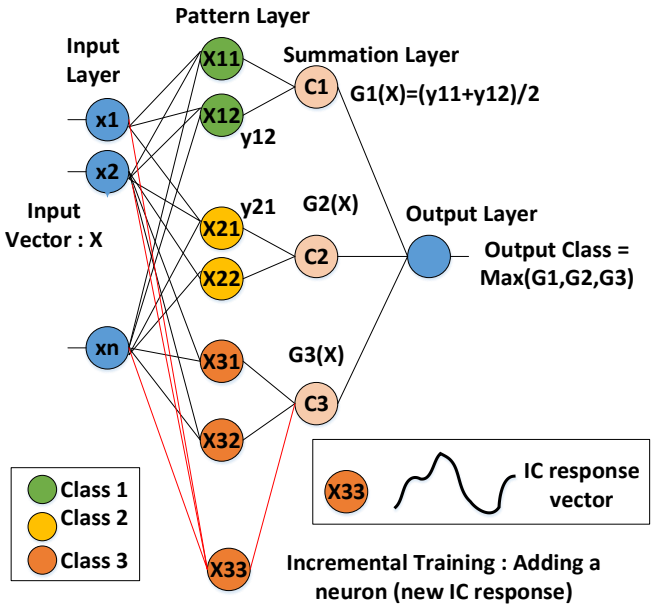


Figure 10 : PNN classifier

In this work, probabilistic neural network (PNN) is used as a classifier. As shown in Figure 10, PNN consists of four layers: i) input layer ii) pattern layer/training Set iii) summation layer and iv) output layer. Each training set data becomes a neuron in pattern layer. Each neuron in the input layer contains components of the input vector. Distances between each stored vector and input vector are enumerated in pattern layer. Based on these distances, at the summation layer the following probabilities are calculated (given in Eq. 15):

$$p_i = \Pr(X \in class\ k | observations\ in\ class\ i) \quad (15)$$
$$X: input\ vector$$
$$number\ of\ classes: 1,2,3,\dots k$$

At the output layer maximum of the above probabilities is found and corresponding class is predicted to be the class of the input vector X. Probability density functions (pdf) of the classes are estimated from training samples using Parzen's [35] pdf estimation technique. The pdf of a single sample and of a single population are given by Eq. 16 [36, 37].

$$f_1 = \frac{1}{\sigma} \mathrm{W}\left(\frac{X - X_k}{\sigma}\right) \quad f_{pop} = \frac{1}{n\sigma} \sum_{k=1}^{n} \mathrm{W}\left(\frac{X - X_k}{\sigma}\right) \tag{16}$$

X: Input Vector X_k: k^{th} sample W: weighting function
σ: smoothing parameter ($0<\sigma<1$)

For a Gaussian weighting function and for an input vector of length p, pdf of samples and class are given by Eq. 17 and 18respectively.

$$\frac{1}{(2\pi)^{p/2}\sigma^p} \exp\left(-\frac{\left\|\vec{X} - \overrightarrow{X_k}\right\|^2}{2\sigma^2}\right) \tag{17}$$

$$G_i(X) = \frac{1}{(2\pi)^{p/2}\sigma^p n_i} \sum_{i=1}^{n_i} \exp\left(-\frac{\|X - X_{ik}\|^2}{2\sigma^2}\right) \tag{18}$$

There has been research on outlier detector before [38, 39]. The advantage of the proposed one, in comparison to the state of the art are (i) when extra observation points are available we do not need to retrain the network, only incremental training (adding new neurons corresponding to the new observation points) is required. An example is shown in Figure 10, where an extra data X33 is added in the PNN. (ii) Training of PNN is order of magnitude faster than any back propagation Neural Net algorithms. (iii) The proposed outlier filter has the capability of rejecting devices based on threshold probability of

acceptance, if it is not similar to any one of the training classes. Rejection criteria has enabled this outlier filter to handle process shift (discussed in detail in results section). (iv) outlier filters in [38, 39] can screen defective devices from normal devices. The proposed outlier filter can screen out defective devices, can classify normal devices into process bins. Drawback of PNN is that a large amount of memory is required to store all the neurons.

2.5 RF Model Building

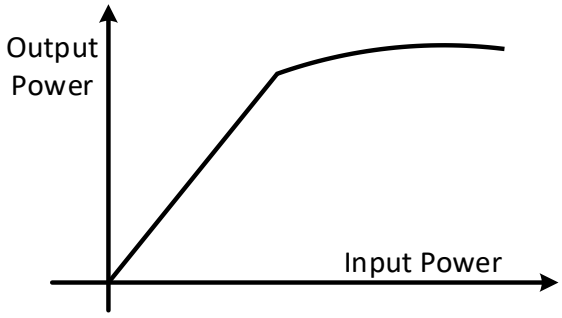


Figure 11 : Input output transfer curve

$$y(t) = \alpha_1 x(t) + \alpha_2 x^2(t) + \alpha_3 x^3(t) + \alpha_4 x^4(t) + \alpha_5 x^5(t) \quad (19)$$

The test stimulus generation algorithms are explained in section 2.3.1 and 2.3.2. In alternate testing approach the specifications are predicted from a signature, build from output response of the DUT corresponding to a well-crafted test stimuli. There are two ways to generate this test stimuli: a) by using actual hardware DUT [16, 17] b) by using a software model of the DUT [40, 41]. Test generation is an expensive process for both the above-mentioned ways. For example, if it is generated on 1000 process varied devices and test generation algorithm runs for 50 iterations then 50,000 simulations (for software models) are required or 50,000 times the DUT needs to be replaced in the load board

(hardware DUT). So, the cost-effective way to do test generation is to build the test on very simplified behavioral model. As shown in Figure 11, by simulating SPICE Netlist or by actuating hardware DUT, the input output power transfer curve is obtained and this curve is modeled as a polynomial (shown in Eq. 19).These models are good enough to capture some basic parameters such as gain, IP3, 1 Db compression point etc. but lacks the exact waveform capturing capability which is required to predict other critical parameters (Bias Current, Bias Voltage etc.).

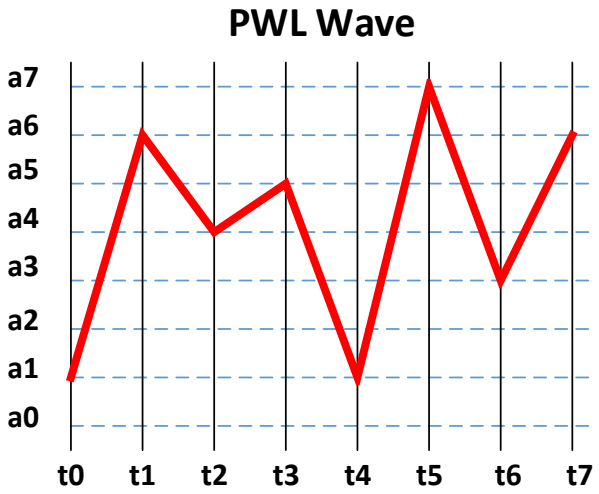


Figure 12: PWL stimulus for alternate testing

In this work, we will generate test on a new software model which is as fast as behavioral model in simulation and as accurate as SPICE simulation for test generation purpose. In test generation, some information such as range of input frequency, range of input slopes etc. are known a-priori. In [42] and [43] the authors have explained a novel Boolean model for linear and non-linear circuits respectively. We have taken this modeling approach and customized it to model any analog/RF envelope. In this work the test stimulus we are trying to come up for stimulating the devices is a PWL wave of finite duration (as

shown in Figure 12). Let's assume that the dynamic range of input stimulus be [0 to V volt] and time duration be T second. T is equally divided so that the adjacent time instants are separated by δt second. For time instants $t_0, t_1, t_2 \ldots\ldots t_k$ we need the amplitude values $v_0, v_1, v_2 \ldots v_k$ to craft the desired PWL wave. These amplitude values $v_0, v_1, v_2 \ldots v_k$ are the output of an optimizer (test generator) which tries to find best possible combination of these amplitude values to optimize some given cost function. To evaluate this cost function, the genetic optimizer needs to evaluate circuit output response repeatedly (based on the model given to it) for every iteration and for every process varied device.

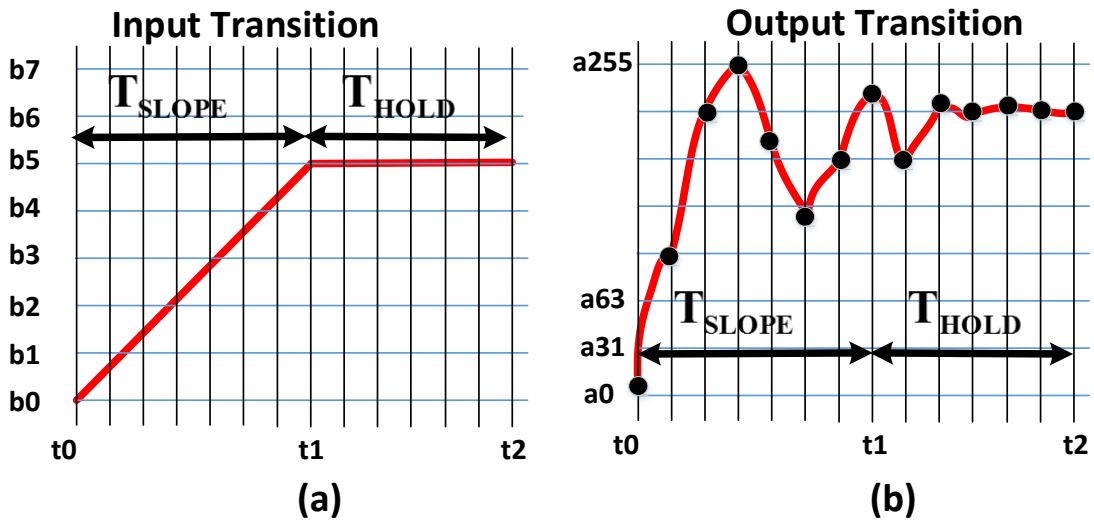


Figure 13: (a) Input transition (b) Output capturing for an input state transition

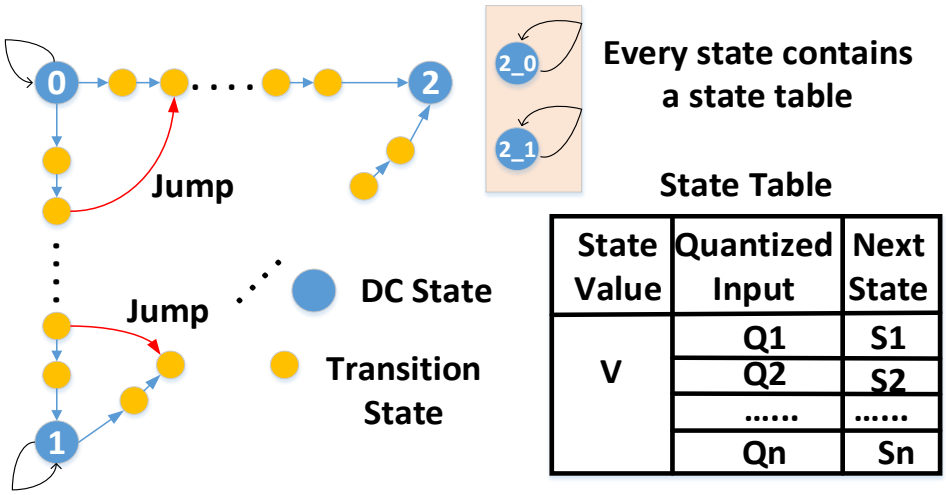


Figure 14: State transition graph

For Boolean model generation (as shown in Figure 13) the input dynamic range and output dynamic range are quantized in 16 and 256 levels respectively. In this way we restrict our input transition to 256 ways. Corresponding to each input level there is a state in the state transition table. These states are known as DC states. Each possible transition is simulated (or captured by actuation for hardware DUT) and captured with 10X sampling rate (as shown in Figure 13b). For all possible input transitions these output response curve is obtained. The output is always sluggish so it may not stable as soon as input is stabilized. From all the transitions the maximum extra time required for stabilizing the output is monitored. Let's assume it is 120% of input duration ($t_1 - t_0$), then for 10X oversampling there will be 12 transition states in between the two DC states in state transition table. An example is shown in explaining state transition if input changes when output is transiting from one DC State to another. As shown in Figure 14, output was transiting from state 0 to state 1 and in between the input changes. This input change corresponds to DC State 2. Now there are two possibilities a) take transition arc 0 to 2 (if present state is close to 0) b) take transition arc 1 to 2 (if present state is close to 1). While jumping from one transition

state to another transition state, continuity of output waveform is maintained. For example, if jumping from 0 to 1 arc to 0 to 2 arc jump to that state of 0 to 2 arc whose output value is closest to the output value of present state in 0 to 1 arc. Accuracy of this Boolean model for a random input is shown in Figure 15. This Boolean model simulation is basically a finite state machine traversal (with proper jump from one state to another based-on input), so inherently very fast in enumerating output envelope. Table 3 is showing a runtime comparison among various plausible simulation choices w.r.t the proposed Boolean model.

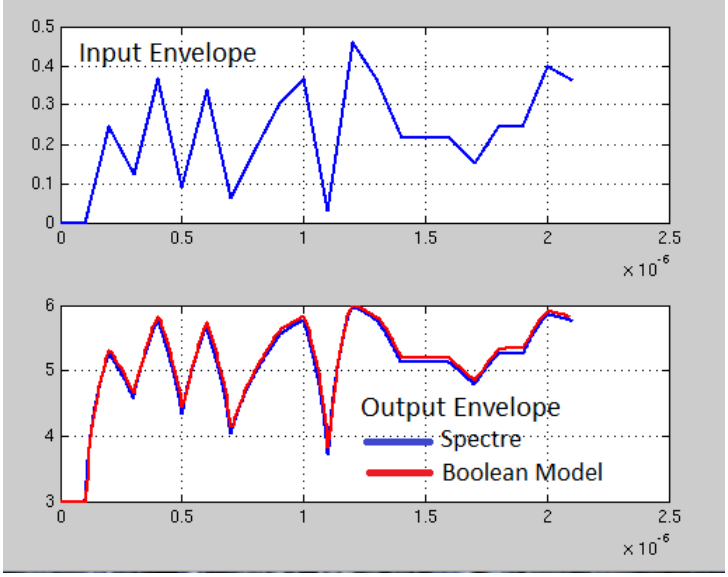


Figure 15: Boolean Model Accuracy

Table 3: Runtime comparison

Boolean Model	Cadence SpectreRF Transient Simulation	Cadence Harmonic Balance Envelope Simulation[44]	HP ADS Transient Simulation
1 ms	4000 ms	500 ms	3000 ms

A pseudo code of the RF model generation is shown in Algorithm 5 and a FSM representation is shown in Figure 14. Every major step is discussed in detail below.

Algorithm 5: Model extraction pseudo code

```
1.   Input :
     i.  Circuit Netlist and Simulator (or Actual Hardware and
     Tester)
     ii.  Number of Input and Output Quantization Levels
     (QLin,QLout)
     iii. Sampling Rate (ts)
     iv.  Tslope, Tstay (see Figure 13)
2.   Output : Boolean Envelope FSM Model

3.   fsm=Createfsm( QLin)
4.   /* Create an empty FSM as shown in Figure 14.Adding only DC
     states to the FSM */
5.   Narc=(QLin(QLin -1)  // Number of Transition Arcs

6.   /*specify state values either from simulation or hardware
     stimulation*/
7.   For i=1 to Narc
8.       If(Simulate_Model==1)
9.           Env_response=Simulate(Circuit Netlist,Input Transition
             Waveform)
10.      else
11.          Env_response=ActuateHardware(IC,Input Transition
             Waveform)

12.      Tsettle=Findsettlingtime(Env_response)
13.      // find settling time for the state transition (see Figure 13)
14.      Env_Response=Env_Response(0 to Tsettle)
15.      //Ki: number of intermediate states in transition arc i
16.      Ki= Env_Response/ts
         createInterMediateStates(I,Ki,Env_Response,ts)
17.
18.  // specify state transition
19.  For i=1 to total number of states
20.    For j=1 to Qin
21.      S_j=findNextState(i,j,Env_response)
         /* finding the next state based on input and analyzing the
22.      envelop responses captured */
23.      Populatefsm(S_j,fsm)
24.      // populate the table shown in  Figure 14 for all the states
```

2.5.1 RF FSM Model Generation:

In ABCD [42, 43], the FSM states correspond to specified transient as well as DC steady state values of the analog circuit inputs with circuit output values associated with every state. Transitions between states are defined by input transitions between quantized levels of the input dynamic range. Since RF circuits are DC-blocking, the above definition of the states of analog circuits needs to be modified to allow RF FSM model generation. In [16] it is shown how the non-idealities of an RF device map onto the baseband signal obtained by RF demodulation. In this work, we assume that an envelope detector is used for this purpose. In our algorithm, *each DC state of the FSM represents the magnitude (quantized) of the baseband signal* and is associated with the corresponding output value of the *demodulated* (baseband) signal (see Figure 2).

2.5.2 Incorporation of Memory Effects

Given the above descriptions (transient, DC) of the states of the FSM model, ABCD constructs transitions between the states of the FSM model by directed Spice (transient) simulation. While a transition is in progress from one DC state to another, it is possible that the input experiences a transition before the circuit output has settled to its new value. This is handled in ABCD [42, 43] using transient "jump" states that define the sequence of transitions the circuit output goes through while "moving" from one DC state to another. To capture such jump states *the complete transition between the respective DC states needs to be simulated*.

In ABCD-RFH, the FSM model and the DC/transient states are created by replacing Spice with direct hardware stimulation. However, because the baseband data

rate is known, the DUT need not be stimulated beyond the duty cycle of the input data rate while transitioning from one DC state to another. However, this creates a problem. As opposed to ABCD in which all transitions from other states to a known DC state result in the *same* output value, in ABCD-RFH, because the DUT is stimulated only within the duty cycle of the input data rate, transitions from other states to a known DC state can result in *different* output values. To accommodate this, we *split the corresponding DC state into multiple DC states* depending on how that state is reached from other FSM (DC) states.

2.5.3 Extraction of FSM Models from Hardware

By replacing Spice simulation with directed hardware stimulation, the cost of model generation incurred by ABCD is virtually avoided. Moreover, these models can be used to simulate how devices behave *even after the latter have been shipped to customers*, a capability that *does not exist with any other tool today*.

The hardware setup for this experiment is shown in Figure 16. In this experiment Device under Test (DUT) is a LNA (X3533 7AZCYH3) mounted on a load board .A high frequency signal generator is used to generate LO signal (2.4GHz -10 dBm). Low frequency envelop signal is generated by Labview and imparted into the circuit by NI data cards. The modulated signal coming from up conversion mixer is used to stimulate the LNA. Output envelope of the LNA is captured by an envelope detector (ADL 5511). Envelope detector output voltage and rms power output constitute signature of the device in this work. Envelope detector output is feed to NI digitizer and data is captured. From extracted data, envelope models are generated by Matlab. For model generation the input slope and the minimum time required to hold it at the current level to stabilize the output

response is critical (shown in Figure 13b). If the model is generated for input slope T_slope then in test generation no transition steeper than T_slope is permissible. This input slope in hardware is constrained by DAC capability. Hold time (T_stay) cannot be predicted a priori so some repeated trials are required. In this hardware experiment T_slope and T_hold are taken as 1e-5 and 4e-4 second respectively. Accuracy of the FSM mode for a random baseband input is shown in Figure 17.

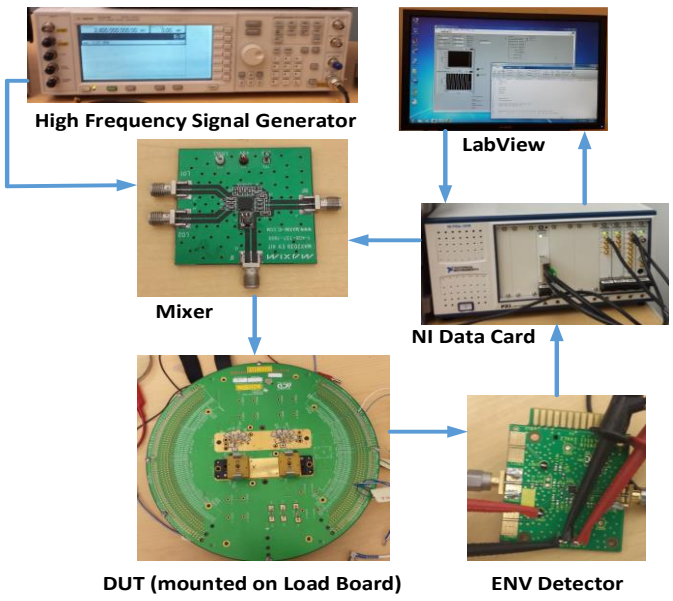


Figure 16: Hardware experiment setup

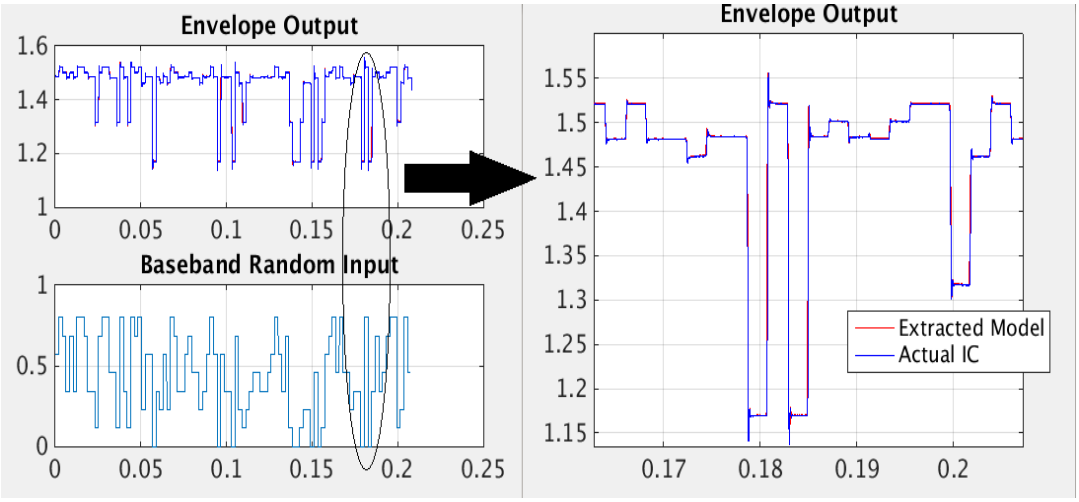


Figure 17: Extracted FSM model accuracy

2.6 Simulation Results

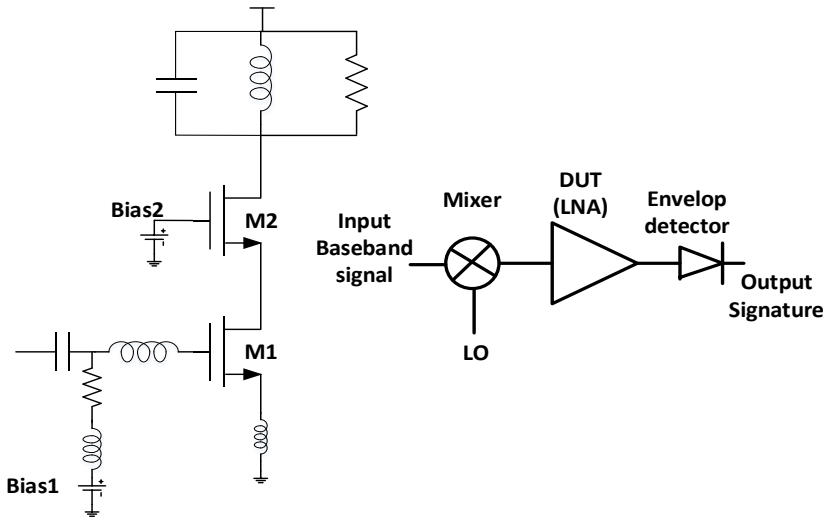


Figure 18: LNA as design under test

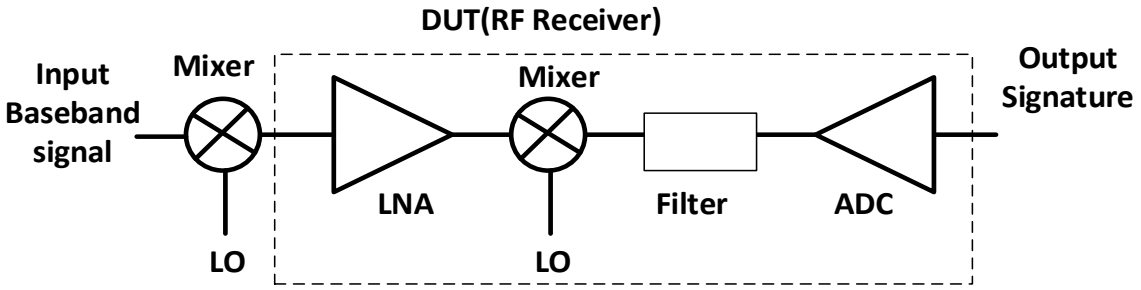


Figure 19: RF receiver as design under test

To show the efficacy of the proposed test methodology, simulation results for an RF circuit (LNA shown in Figure 18) and an RF system (RF receiver shown in Figure 19) are presented here. Process parameters (vth, tox, W, L) were varied in Gaussian fashion to create three process lots ($\sigma = 5\%, 10\%\ and\ 15\%$ of mean value). In this work 45nm FreePDK models [45] are used in simulation. Following the sampling criteria mentioned in section V, number of ICs sampled for LNA and receiver system and pass/fail criteria of the developed tests are given in Table 4. Specification distribution of sampled devices are shown in Figure 20 (Gain and IIP3 for LNA) and Figure 22 (EVM for RF receiver).

Table 4: Test specifications

DUT	# ICs used for model building	# ICs used for validation	Pass/Fail criteria
LNA	6000	2000	Gain> 26.5dB IP3> 7.5dB.
RF Receiver	10,000	4000	EVM<5%

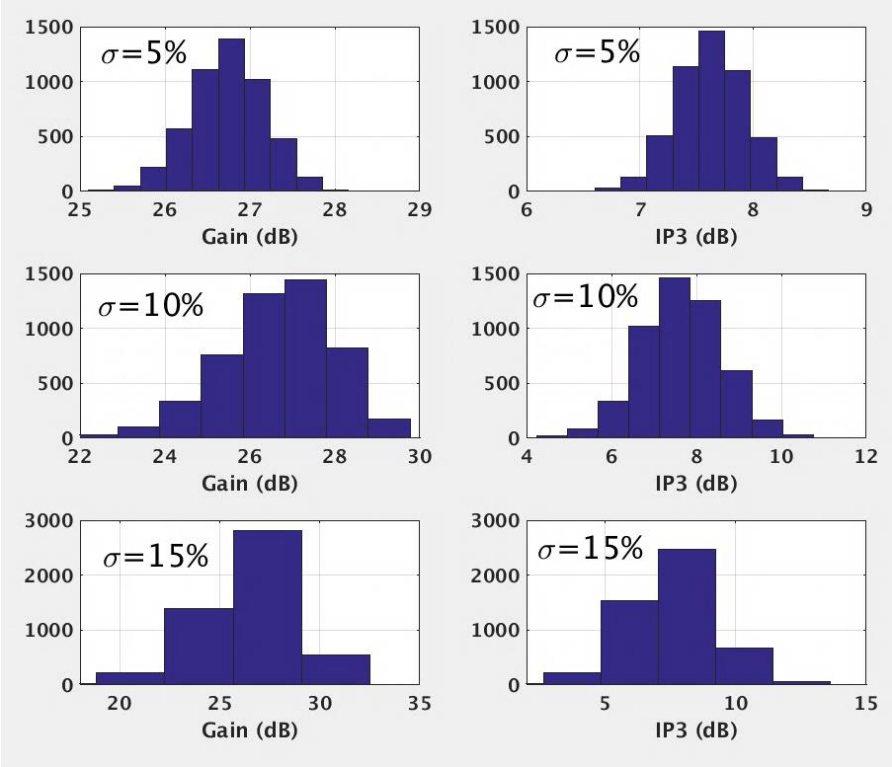


Figure 20: Gain and IIP3 distribution of sampled devices

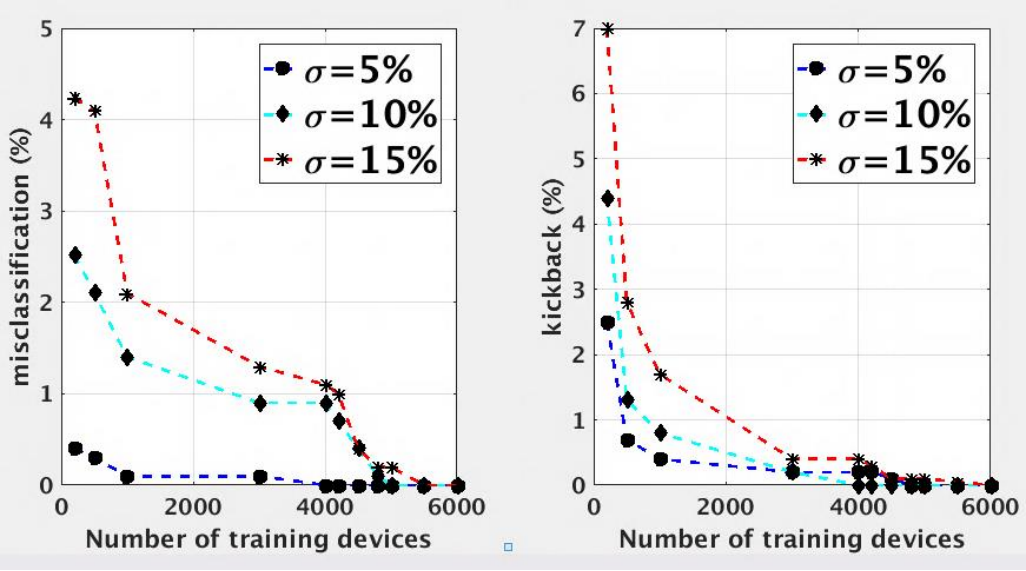


Figure 21: Misclassification and kickback vs number of training devices (LNA)

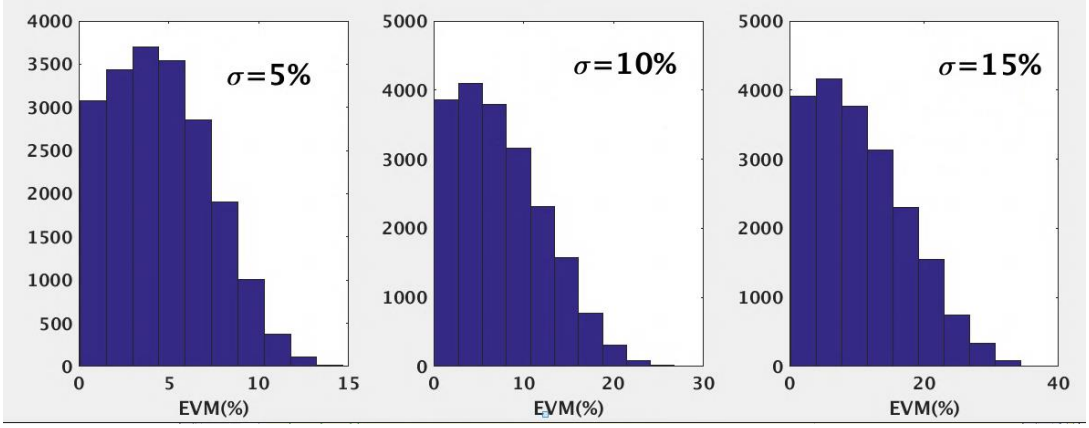


Figure 22: EVM distribution of sampled receivers

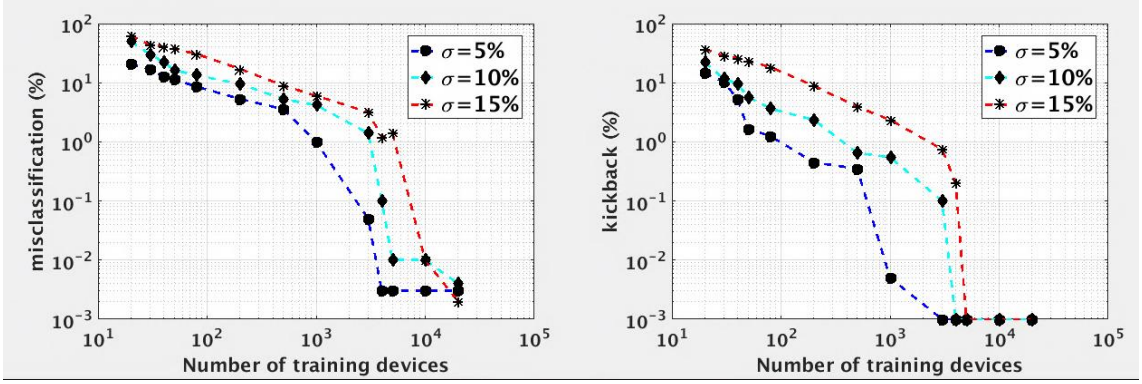


Figure 23: Misclassification and kickback vs number of training devices (Receiver)

Following the procedure of section 2.3 and 2.5 , FSM models of these devices and tests are created. How the test and model is evolved are shown in Figure 21 and Figure 23. It is apparent from these two figures, that higher the process variation is, more the number of devices required to build and tune the model. In this work a model and test are said to be ready for production testing when misclassification and kickback rates are below 0.1%. Number of LNA ICs required to complete the model for three process lots are 4000, 4800 and 5500 respectively. Number of receiver ICs required to complete the modeling for three process lots are 5000, 6000 and 8500 respectively.

To show the efficacy of the classification performed by PNN, we compared the performance of the PNN with two similar classifiers K-Nearest-Neighbors (KNN) and multinomial logistic regression. Like PNN, KNN also supports incremental training, adding or removing observation data is simple in KNN. But it does not provide threshold probability confidence as in PNN. On the other hand, multinomial logistic regression does provide threshold probability confidence as PNN, but does not support incremental training. Table 5 shows run time comparison among the different classifiers for LNA ICs of process lot 1. Classification performances of different classifiers are shown in Table 6. Matlab functions newpnn, fitcknn and mnrfit are used to build the respective classifiers.

Table 5: Run time comparison (secs) for LNA ICs in process lot 1

PNN	KNN (K=4)	Multinomial logistic regression
0.14	0.08	40.2

Table 6: Misclassification rate for LNA ICs in process lot 1

PNN	KNN (K=4)	Multinomial logistic regression
0.01 %	0.1%	0.2%

2.7 Hardware Measurement Results

Using the hardware setup described before, we initially extracted FSM model for 2500 devices. 1500 devices we use for model building and rest 1000 devices are used for validation purpose. Initially a crude model is built from 300 devices. The outlier filter rejected almost 40 % of the validation devices (Figure 24). As shown in Figure 24, beyond 300 sample devices incorporated in outlier filter the rejection rate starts decreasing and

after 800 devices the model becomes accurate enough for production testing. Misclassification rate of the outlier filter also gradually decreases and after tuning with 800 new devices (in excess to initial model built with 300 devices) the outlier filter becomes suitable for production testing. Better the outlier filter is, higher will be the accuracy of specification prediction for MARS mapper. This is corroborated in Figure 25 where specification prediction accuracy for each specification is shown at three model building phases. Based on outlier filter response the MARS model and stimuli are also tuned to improve the test. It is argued before that higher the correlation between alternate measurement and specification values better would be the prediction. How the correlation at initial model building stage and after model tuning changes are shown in Figure 26 (a).

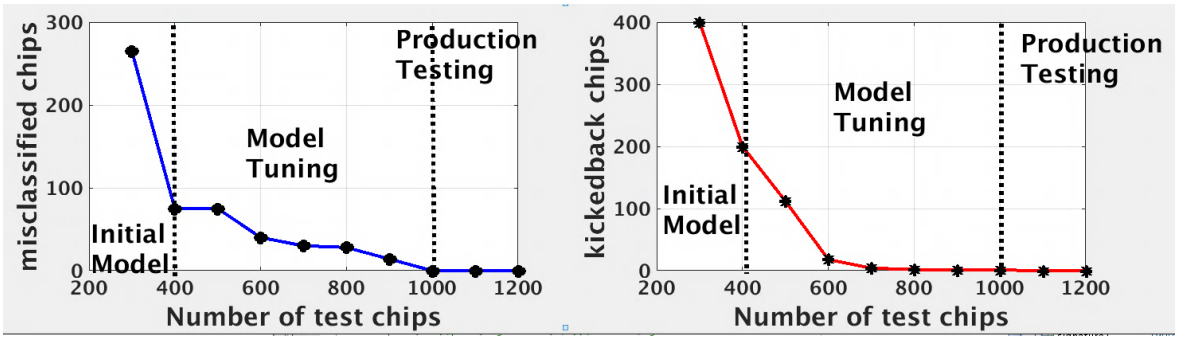


Figure 24: Misclassification and kickback rate at different phases of testing

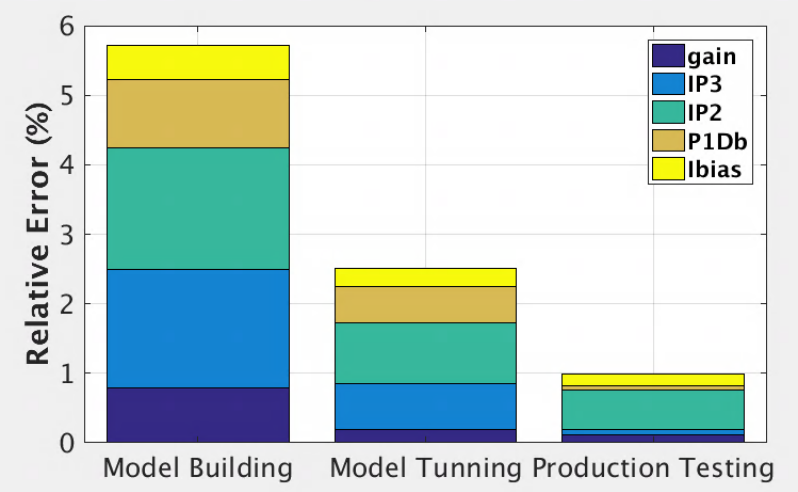
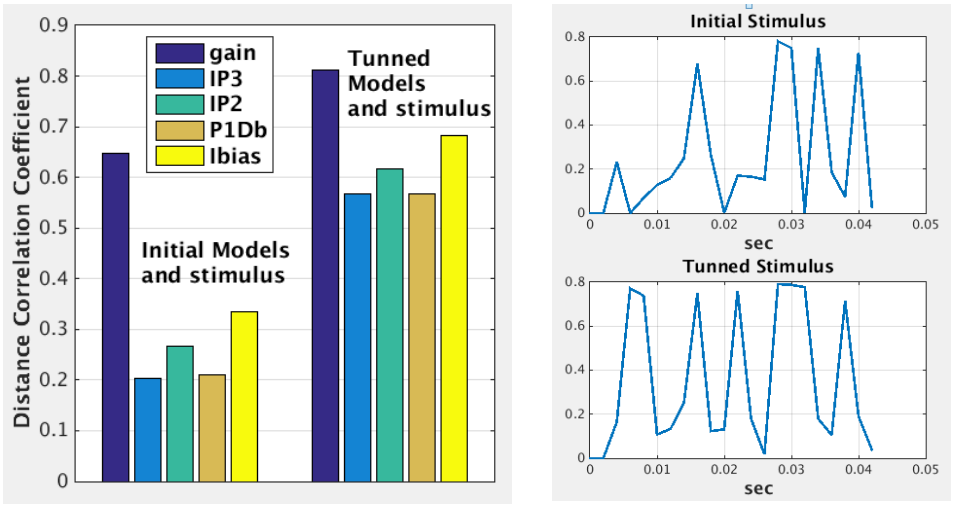


Figure 25: Relative error in specification prediction at different phases of test

We ran another interesting experiment (in simulation) with a LNA where we change process lot. Initially for process lot 1, the outlier filter, regression mapping functions and test stimulus are tuned to production testing level accuracy. It required almost 800 devices to go to production test accuracy level (both rejection ratio and misclassification rate below 0.5%). After that any device coming from process lot 1 (Figure 27) will be correctly classified and its specification can be accurately predicted from the test. What would happen if any device from other process lot comes? As the outlier filter has no information about that process space it would reject all these devices initially as shown in Figure 27 (abrupt change in rejection rate). As mentioned in section II, all outlier filter rejected devices are kicked back to standard testing and their results are incorporated into test procedure. The test procedure is tuned after every 100 devices until it stabilizes. For process lot 2, only 400 devices stabilize the test procedure as opposed to process lot 1 where 800 devices were required. This is due to the reason that process lot 2 learns from already available process lot 1 information.



(a) **(b)**

Figure 26: (a) Correlation before and after model tuning (b) optimized stimulus

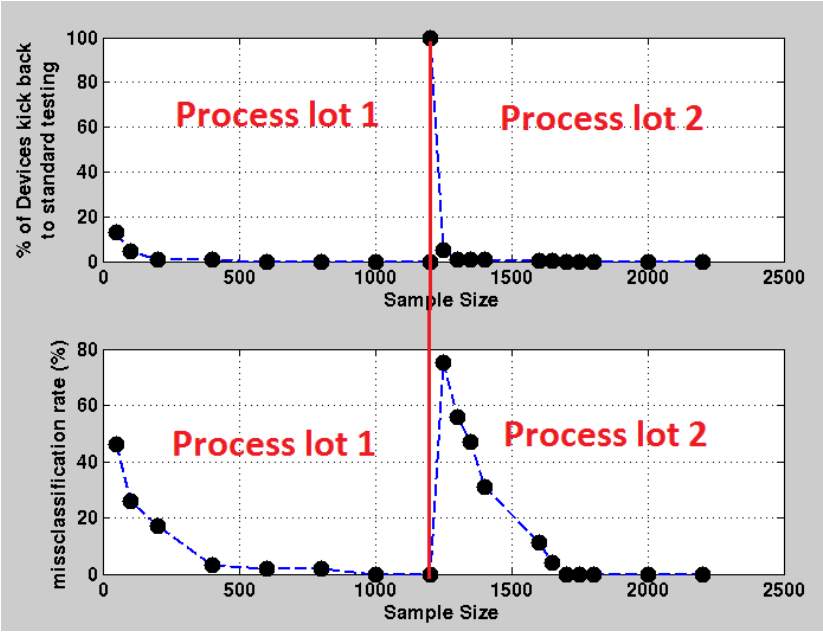


Figure 27: Test development for process shift

2.8 Conclusions

In this work, we have described a novel adaptive testing methodology for RF/analog ICs. The proposed procedure is validated in simulation and on TI transceiver ICs (X3533 7AZCYH3). FSM models capable of ultra-high speed transient simulation,

along with niche stimulus generation algorithms have achieved more than 100X speedup in stimulus synthesis. Adaptively tuning parameters of the testing procedure quickly converges (with less number of sample ICs) the initial crude parameters to production testing accuracy level. In our future work we would like to extend this adaptive testing to RF/analog systems.

CHAPTER 3. VAST: POST-SILICON VALIDATION AND DIAGNOSIS OF RF/MIXED-SIGNAL SYSTEMS USING SIGTATURE TESTS

3.1 Introduction

Validating the correct operation of RF/Mixed signal circuits is getting increasingly difficult due to increase in electronic system complexity, rapid technology change and decreasing design cycle time. To ensure performance reliability, reduce yield loss and to reduce design and test cost there is need for automated validation/diagnosis methodology for electronic systems and circuits. Technology scaling has enabled dense integration of digital and analog/mixed signal functionality in the same die area making the problem of testing and validation of mixed signal designs even more difficult. There is an effort in digital design, especially in microprocessor design [46-48] and memory subsystem [49] to develop aggressive post silicon validation methodologies geared towards finding behaviors in silicon that are difficult to model a priori . Of specific interest are electrical bugs due to signal coupling, ground bounce, substrate noise and other higher order effects that are difficult to include in digital simulation algorithms. However, validation of mixed-signal/RF systems for un modeled higher order effects is difficult because precise simulation of all electrical aspects of the design considering interfaces between digital analog and RF circuitry, including the effects of process uncertainty etc. is difficult and computationally expensive. The problem is expected to get worse beyond 90nm technology nodes for mixed signal/RF systems. In this context, post-silicon validation of mixed-signal SoCs for pre-silicon design verification “escapes”, particularly “escapes” related to

electrical bugs, is rapidly becoming imperative for advanced high-speed designs and is a key challenge.

3.1.1 RF/Analog Circuit Design Anomalies Classification

With regard to validation, we are primarily concerned with design errors that lead to unexpected device behavior. There are two broad classes of such design anomalies:

(a) Due to unmodeled design effects in silicon. These models are defined to be "incomplete". (b) Due to models that capture all relevant behaviors, i.e. are "complete", but whose design parameter values do not match with fabricated silicon.

In (b), there can be "hard" design errors in which the silicon parameters are considerably different from the model parameter values or "soft" design errors for which silicon parameters are marginally different from the model parameter values. The objective of design validation is to first prove equivalence between the DUT model and the observed DUT behavior in silicon and then find the source of the design error in case it is determined that the DUT behavior is different from that predicted by its model. In fact, design validation techniques can be used to detect the effects of process variations [50] and aging for which the threshold voltage of devices changes over the lifetime of device operation due to NBTI and PBTI effects in nanometer nodes [51]. Process variations and aging change the circuit parameter values causing equivalent "soft" design errors.

3.1.2 State of the Art Mixed Signal/RF Verification/Validation Techniques

State of the art mixed-signal/RF verification techniques are described in [52, 53] and involves accurate behavioral modeling, fast simulation and simple property checking

(assertions inserted by the designer). Also, simulations are performed to ensure that the design specifications are not violated in the presence of process variations. However, while a framework for mixed-signal verification is given, the detailed validation requires human input and knowledge. In [52, 53] regression tests are used to verify mixed-signal behavior. However, generating the tests in an automated manner is a significant problem. Analog hardware description languages (Verilog AMS, VHDL AMS), and use of MATLAB/Simulink for modeling of mixed-signal SoC designs are described in [54-58]. There has also been research on formal verification of mixed-signal circuits based on property checking [59, 60]. However, it is difficult to check the dynamic behavior of mixed-signal/RF systems using formal checking methods. An effort in this direction was made by the authors in [61, 62]. A key contribution here was to formulate the design verification problem as an optimization problem to determine combinations of design parameters that could result in one or more specification violation. In contrast, there has been little work in the area of post-silicon debug of mixed-signal/RF systems. Post-silicon debug is necessarily driven by tests applied to the manufactured DUT and by back-end algorithms that diagnose the cause of anomalies in observed design behavior. It is complicated by the fact that is possible for mixed-signal circuits to meet their design specifications but still exhibit spurious/malicious behavior for specific inputs due to electrical bugs that are difficult to simulate pre-silicon. In the digital space, there has been significant work in test generation driven verification [50, 51, 63, 64] and in the use of specific programs for post-silicon validation and design debug [46, 47] . However, there has not been significant work in test generation driven post-silicon validation of analog/mixed-signal/RF circuits and systems. In this chapter we will discuss two signature

based validation techniques (a) model parameter tuning based validation (b) learning assisted validation.

3.2 Model Parameter Tuning Based Validation

Post silicon debug/validation of RF and mixed signal circuits is predicated on a model (behavioral or circuit-level at the highest possible level of detail) which describes the desired behavior of the DUT. As mentioned before, due to technology and speed scaling of mixed-signal/RF devices, it is difficult to incorporate all the effects of electrical bugs in the DUT model. The validation problem consists of determining if there are electrical bugs in silicon that cause the DUT to exhibit behaviors not predicted by the DUT model. In this work, our objectives are as follows:

(a) Determine if the DUT contains behaviors not defined by its underlying model. This is the same as determining if the model is "complete" as described earlier.

(b) If the model is complete, but validation testing determines that there is still a discrepancy between the expected (produced by the model) and the observed DUT behavior, then how does one localize/diagnose the discrepancy observed in the DUT behavior down to a specific design module?

(c) Through validation testing, if the model is determined to be not "complete", is it possible to update the model so that the observed (unexpected) behaviors can be explained with high confidence?

In this work, a specially designed test derived from consideration of the DUT model is used to *simultaneously* stimulate the DUT and its model (running on an

emulator/simulator). Any difference between the DUT response and the model response is treated as a *design anomaly signature*. If this difference cannot be brought down to a value below a specified threshold (defined by simulation accuracy) by perturbing the model parameters using known optimization methods for minimizing this error, then the obtained DUT response cannot be explained by its model and we conclude that the model is *incomplete*. We then invoke *model update* procedures and repeat the validation step until the model is determined to be complete. Under the assumption that the design anomaly is due to a single embedded DUT module, the approach proposed in this chapter determines if the model is complete and also the specific embedded module that caused the anomaly in a *single step*. This anomaly isolation technique for single DUT module malfunctioning is computationally inexpensive. If the model is found to be incomplete, a heuristic model update procedure is described in this work and demonstrated for an RF front end. This work is organized in the following way: Section 3.2.1 describes the test generation approach for a transceiver. Section 3.2.2 describes a model completeness checking algorithm to determine if the DUT contains behaviors not defined by its underlying model. Section 3.2.3 describes the anomaly isolation procedure, under the assumption that the model is complete, but model parameters are off from nominal values. Section 3.2.4 describes model update heuristic if the model is found incomplete. The transceiver model used for simulation and validation is explained in section 3.2.5.

3.2.1 Test Signal Generation for RF Transceiver

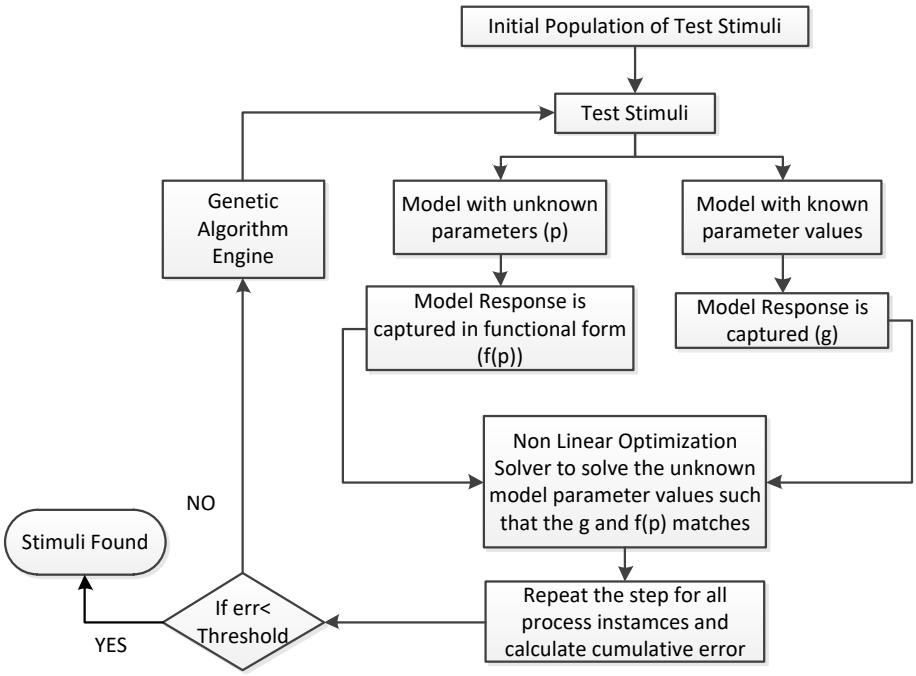


Figure 28: Test generation algorithm for RF transceiver

From nominal device parameters, Monte Carlo sampling is used to create different process instances. We have created 300 malfunctioning transmitters where 100 have a malfunctioning PA, 100 have a malfunctioning I-Mixer, and the remaining 100 have a malfunctioning Q-Mixer. Based on the model, an input stimulus is generated which will be able to excite all the non-idealities (all the non-idealities present in all the modules) present in the model. The test stimuli are 64 tone signals used to transmit OFDM symbols. The detailed test generation algorithm can be found in [65] and [66] . Similarly, the test stimuli for the receiver are also generated. It is to be noted that test generation is only possible when model of the device is known. For un-modeled effects test generation is not possible, we have to rely on huge set of random stimuli. The test generation algorithm starts with an initial population of test stimuli. These stimuli are applied to the model, and model

response is captured. From this model response we use a nonlinear optimizer to back calculate the model parameters so that the error between two model responses (g and f in Figure 28) is minimized. Proper back calculation ensures that the stimuli have excited all the effects present in the model. This step is repeated for all the process instances and cumulative error function is calculated which is used as an objective function for the genetic optimization engine. Objective function of the GA is given in Eq. 20.

$$minimize \quad \sum_{i=1}^{n} \left(p_i - p_i^{estimated}\right)^2 \tag{20}$$

Where $p_i \; and \; p_i^{estimated}$ denotes parameter values and estimated parameter values from non liner regression solver and objective function for non-linear regression solver is given in Eq. 21.

$$minimize \; f(p) - g \tag{21}$$
$$where \;\; f \; is \; the \; model \; function \; p \; is \; the \; parameter \; value \; to \; be \; estimated \; and \; g \; is \; the \; circuit \; response$$

3.2.2 Model Completeness Checking

Figure 29 shows the model completeness checking procedure to determine whether the model is capable of describing the DUT behavior or not. The stimulus obtained in section 3.2.1 is applied to both the DUT and the model, and their responses are captured. An error metric is calculated based on their response signatures. If error metric is below a threshold level, we infer DUT behavior is completely described by the model and the model is complete. If the error metric (shown in Eq. 22) is above the threshold level, then DUT behavior is not completely described by the model. This mismatch can occur due to two

reasons. (a) The model is incomplete or (b) The model is complete but the model parameters have deviated due to process variations or design errors.

$$Error\ Metric = \sum_{t=1}^{N} |f(p) - g|^2 \tag{22}$$

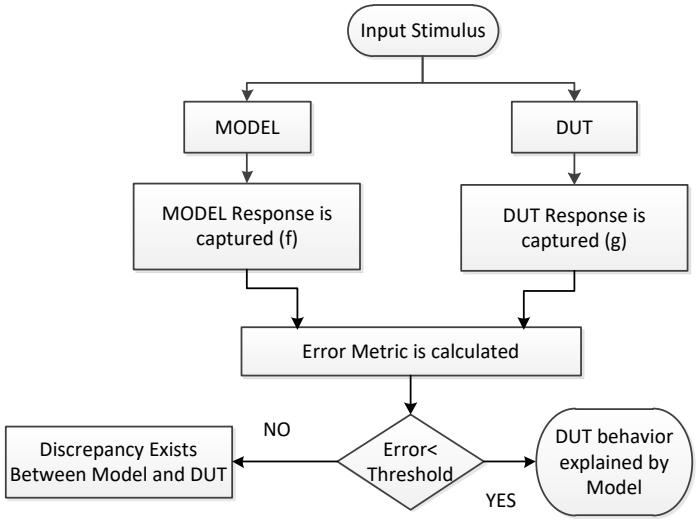


Figure 29: Model completeness checking procedure

3.2.3 Anomalous Behaviour Isolation

In previous section if the model and DUT are found to be non-equivalent, then the next step in validation is to find whether the model is complete and to localize any anomalies between the DUT and the model to specific DUT modules. The proposed approach of this work does these two (model completeness checking and anomaly isolation) in a single step.

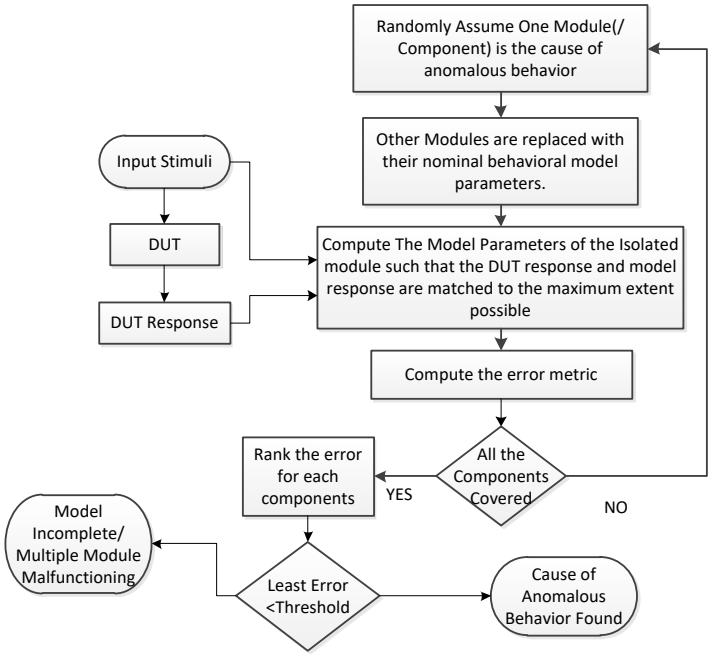


Figure 30: Anomaly isolation (single module malfunctioning assumption)

The algorithm for isolating the root cause of behavioral mismatch (under single module malfunctioning assumption) is depicted in Figure 30. A replica of the RF circuit model is taken. Initially, one component is randomly assumed to be the source of anomalous behavior, and other components are replaced with nominal parameter values in the replica circuit model. By a nonlinear optimization technique, model parameter values are predicted such that the DUT signature is matched with the model signature to the maximum possible extent. This process is repeated, changing the location of the assumed anomalously behaved module over all the components in the device, and the corresponding error metric values are ranked. If the least residual error among the modules is below the threshold limit then the module corresponding to the least error metric value is considered the cause of mismatch. For the RF transmitter and receiver the error metric is given in Eq. 23 and 24.

$$Error_{transmitter} = \sum_{t=1}^{T} \left| ENV(t) - ENV_f(t) \right|^2 \quad (23)$$

$$Error_{receiver} = \sum_{t=1}^{T} \left| IR(t) - IR_f(t) \right|^2 + \sum_{t=1}^{T} \left| QR(t) - QR_f(t) \right|^2 \quad (24)$$

$ENV(t)$ *envelope outputresponse of the replica circuit*
$ENV_f(t)$*envelope output response of the faulty circuit*
$IR(t)$ $QR(t)$ *in phase and quadrature output of the replica circuit*
$IR_f(t)$ $QR_f(t)$ *in phase and quadrature output of the faulty circuit*

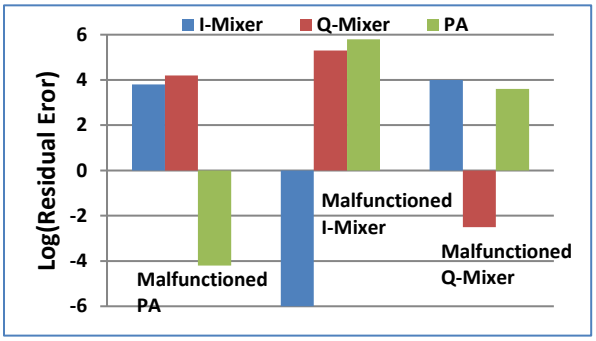


Figure 31: Transmitter discrepancy localization

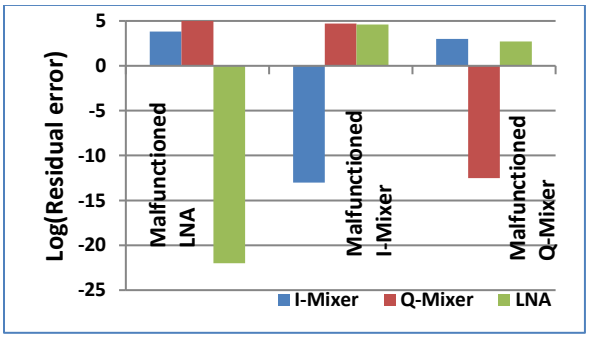


Figure 32: Receiver discrepancy localization

Figure 31 and Figure 32 depict the discrepancy isolation technique discussed above. Figure 31 shows the residual error values for three different anomalously behaved devices. In the 1st, 2nd and 3rd cases, the PA, I-Mixer, and Q-Mixer respectively, are malfunctioning. In the first device where PA is malfunctioning, trying to match the DUT and model

responses by assuming PA is malfunctioning and solve for PA parameters (assuming I Mixer and Q Mixer are behaving in accordance with the model) can yield a good match. But beforehand we do not know which sub-module (I Mixer, Q Mixer, and PA) is malfunctioning, so we have to repeat the process for all the three sub-modules and rank their residual error. For the first device (shown in Figure 31) we found that residual error corresponding to PA is far below that of the other two, and we infer PA is malfunctioning in 1st device. Similar results can be observed in 2nd and 3rd devices where I Mixer and Q Mixer were malfunctioning respectively. Figure 32 depicts the discrepancy isolation in receiver. Similar to the transmitter, we have considered three malfunctioned receivers (LNA, I-Mixer and Q-Mixer is malfunctioning in 1st 2nd and 3rd receiver respectively) to show the feasibility of our methodology. In receiver diagnosis, transmitter and receiver are used in loopback. Input stimuli are applied to the transmitter and transmitter output is fed back to the receiver. Receiver output is the observable node. The above experiment is done with 100 of transmitters and receivers and diagnosing accuracy is shown in Table 7.

Table 7: Experimental validation results

Malfunctioned Component/Module	No. of Devices Validated	No. of Devices Correctly Diagnosed
I Mixer(Transmitter)	100	100
Q Mixer(Transmitter)	100	100
Power Amplifier	100	100
LNA	100	100
I Mixer(Receiver)	100	100
Q Mixer(Receiver)	100	100

If the single malfunctioning assumption is failed (not able to bring the residual error below threshold) then multiple module malfunctioning procedure is adopted. Here instead of tweaking single module parameters, all the module parameters are tweaked

simultaneously to match the DUT and model responses. The same optimization technique is used to tweak all the module parameters instead of just a single module's parameters. Although the multiple malfunctioning detection procedure can detect single malfunctioning module, the CPU runtime is higher than that of single malfunctioning detection procedure. CPU runtimes for both the procedures (run on a RF transmitter) are compared in Table 8. Tweaking multiple parameters together not only increases CPU runtime, but it also increases the chance of aliasing and decreases diagnosis accuracy.

Table 8: CPU runtime comparison

Malfunctioning Module	Single Malfunctioning detection Procedure	Multiple Malfunctioning detection Procedure
PA	11.2 sec	16.3 sec
I-Mixer	10.1 sec	16.65 sec
Q-Mixer	11.1 sec	14.2 sec

3.2.4 Model Building Procedure

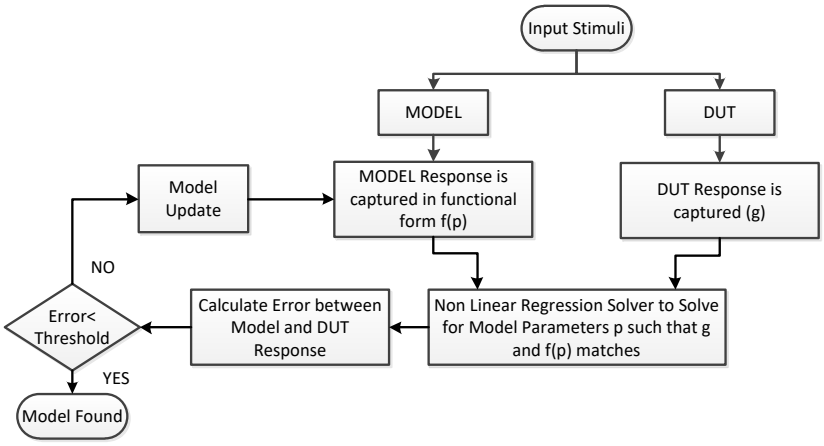


Figure 33: Model building procedure

Figure 33 and Figure 37 shows the model building and iterative model update algorithms. A test generator is first used to generate a test stimulus in such a way that all

the behavioral model parameters of the DUT can be computed from the observed test response. To cover behaviors not included in the DUT model (since we do not know a priori if the model is "complete"), this test is further enhanced with additional randomized stimuli. The generated test is applied concurrently to the DUT (hardware) and its behavioral model with the objective of determining whether the DUT model sufficiently captures observed DUT (hardware) input-output behavior. If a difference is observed, then the next task is to determine if the difference can be explained by perturbing the behavioral model parameters alone. A nonlinear optimizer is used to estimate the model parameters of the corresponding observed DUT response. After solving for the model parameters, model response is captured in numerical from and compared against the observed DUT response and an error metric is calculated. If the error is within the threshold, then the model is adequate to represent the DUT and the model is assumed to be complete. However, if the error is larger than a pre-computed threshold value, then the model is not "complete" and needs to be iteratively updated with repeated test application and diagnosis, as above, until the observed DUT behavior can be explained through qualitative models. Note that if the DUT contains behaviors not included in its model, then no matter how the model parameters are perturbed, the difference between the observed DUT response and its (current) model can never be minimized below a certain value and therefore the model is deemed to be "incomplete".

Model building examples are explained in Figure 34, Figure 35 and Figure 36. Model update is of two types: (a) adding new effects into the model (i.e. incorporating dc offset, IQ gain and IQ phase mismatch etc.), or (b) increasing the complexity level of the

already present effects (i.e. increasing the order of the polynomial capturing AM to AM effect etc.).

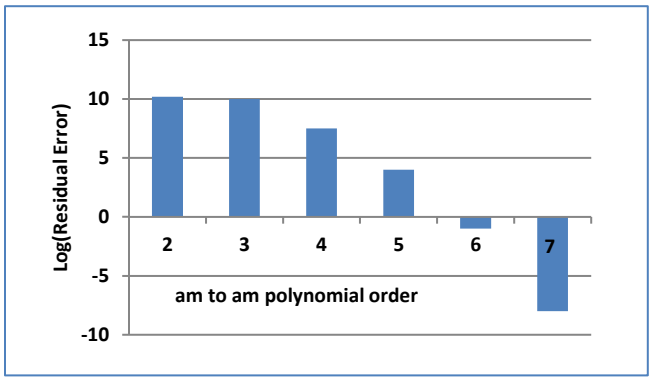


Figure 34: AM to AM model building

Figure 35: IQ mismatch model building

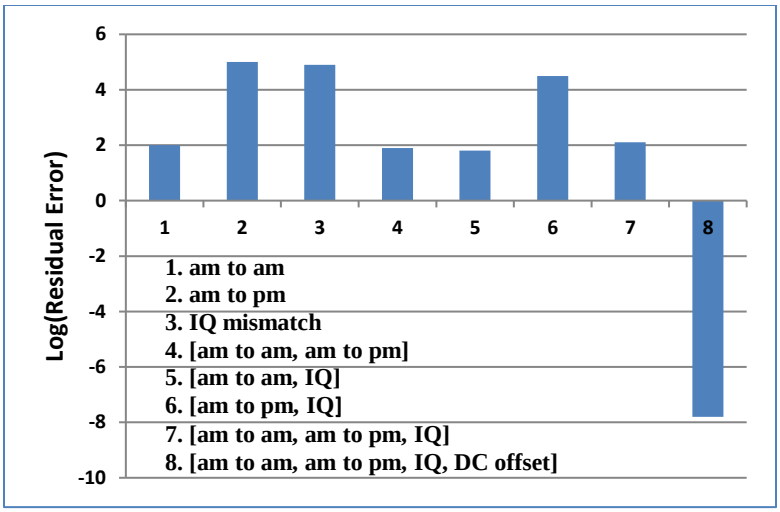


Figure 36: Model convergence

In Figure 34, we show how the residual error decreases as we increase the polynomial order of the AM to AM curve. When model order matches with DUT effects (DUT shows 7^{th} order AM to AM effects) the residual error becomes negligible. In this case, the model contains all the effects shown by DUT only increase in model complexity ensures convergence. Next in Figure 35, we show an example where the DUT exhibit IQ gain and IQ phase mismatch but this IQ mismatch was not included in model. Here as we are increasing polynomial complexity of AM to AM and AM to PM curves for first few increments we got some improvement in residual error but after that no matter how complex the other models are residual error never decreases. This ensures that the model is lacking something which the DUT is exhibiting. In Figure 36 we cite another example where the DUT exhibit AM to AM , AM to PM , IQ gain mismatch, IQ phase mismatch and DC offset effects. Here we show how incorporating these effects one by one reduce residual error and ultimately the model converges when we incorporate all the effects.

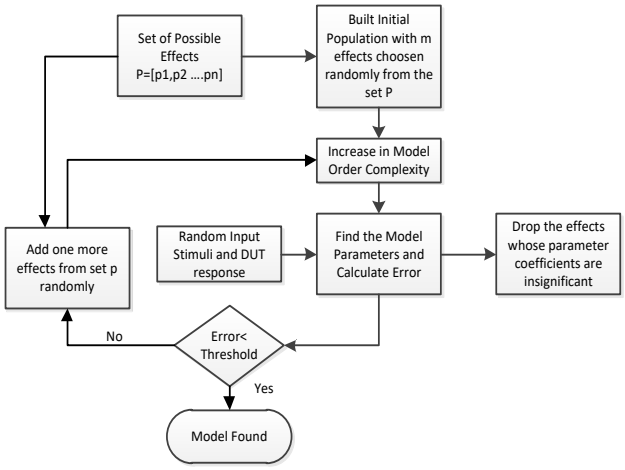


Figure 37: Model update procedure

3.2.5 RF Transceiver Model

RF transceiver model used for simulation and validation is explained here briefly (see Figure 38). A detailed modeling of this transceiver can be found in [63-66]. Transmitter and receiver are connected in loopback fashion for receiver diagnosis. An envelope detector is used at the output of the transmitter to have one more observable node in the system. There are two observable nodes in the transceiver system one at the output of the receiver and the other at the output of the transmitter. At the output of the receiver, down converted in phase and quadrature phase transmitted signals and at the output of the transmitter, envelope of the up converted signal are observed. The transmitter is consisting of Local Oscillator, two mixers, phase shifter, Power Amplifier and filters. This transceiver is suitable for quadrature modulated data transmission. 64 QAM modulated OFDM signal has been used for simulation and validation.

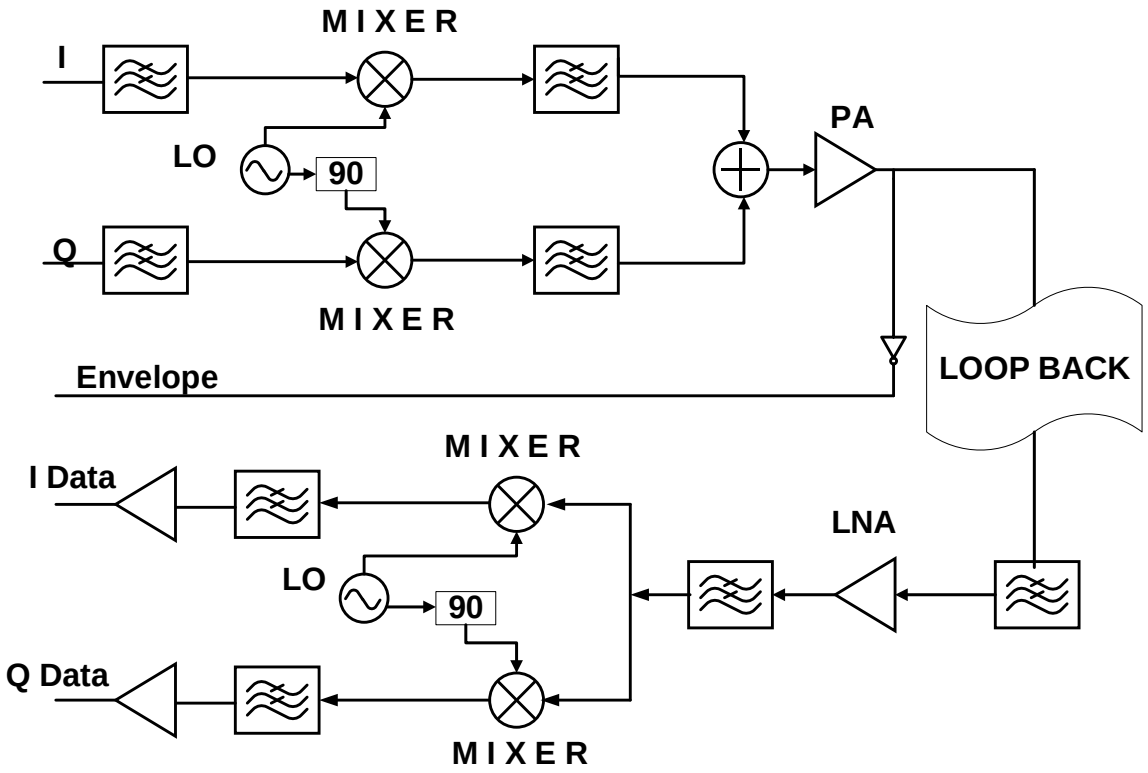


Figure 38: RF transceiver

In phase and Quadrature phase input signal I(t) & Q(t) can be represented as $x(t) = I(t) + jQ(t)$x(t) = I(t) + jQ(t). Mixer amplitude to amplitude distortion, LO phase offset and LO self-mixing/DC-offset are considered in mixer modeling. AM to AM distortion effect is modeled as polynomial of input envelope amplitude (see Eq. 25).

$$g(t) = \alpha_0 + \alpha_1|x(t)| + \alpha_2|x(t)|^2 + \alpha_3|x(t)|^3 + \alpha_4|x(t)|^4 + \alpha_5|x(t)|^5 \quad (25)$$
$$Mixer(t) = g(t)\cos\big(\omega_c t + \phi(t)\big) + \epsilon$$
$$where\ \ |x(t)|: Input\ Signal\ Amplitude\ \omega_c: LO\ frequency$$
$$\phi(t): LO\ phase\ offset\ \epsilon: DC\ component\ Due\ to\ self\ mixing$$

Two mixer outputs are combined through an adder (or substractor) at the input of the PA. Mixer1 and Mixer2 outputs are described in Eq. 26 and 27.

$$MX_1(t) = g(|I(t)|)\cos(\omega_c t + \phi_1) \quad (26)$$
$$MX_2(t) = g(|Q(t)|\cos(\omega_c t + \phi_2) \quad (27)$$

Power Amplifier input signal is given by Eq. 28. On further simplification (using Eq. 29 and 30) Eq. 28 becomes Eq.31.

$$MX_1(t) - MX_2(t) = g(|I(t)|)\cos(\omega_c t + \phi_1) - g(|Q(t)|)\cos(\omega_c t + \phi_2) \quad (28)$$
$$= \cos(\omega_c t)\,[g(|I(t)|)\cos(\phi_1) - g(|Q(t)|)\cos(\phi_2)]$$
$$-\sin(\omega_c t)\,[g(|I(t)|)\sin(\phi_1) - g(|Q(t)|)\sin(\phi_2)]$$

Let $A_1(t) = g(|I(t)|)\cos(\phi_1) - g(|Q(t)|)\cos(\phi_2)$ (29)

$A_2(t) = g(|I(t)|)\sin(\phi_1) - g(|Q(t)|)\sin(\phi_2)$ (30)

$$MX_1(t) - MX_2(t) = A_1(t)\cos(\omega_c t) - A_2(\mathrm{t})\sin(\omega_c t) \quad (31)$$
$$= A(t)\cos\big(\omega_c t + \psi(t)\big)$$

Where $A(t) = \sqrt{A_1^2(t) + A_1^2(t)}$ and $\psi(t) = \tan^{-1}\left(\frac{A_2(t)}{A_1(t)}\right)$

Power Amplifier AM to AM and AM to PM distortion is also modeled as polynomial function of input envelope amplitude. If $x(t)$ is be the input signal to the Power Amplifier then $f(t)$ and $\theta(t)$ describes the AM to AM and AM to PM distortion respectively (given in Eq. 32 and 33 respectively). Power Amplifier output can be described as given in Eq. 34.

$$f_{PA}(t) = a_0 + a_1|x(t)|^2 + a_3|x(t)|^3 + a_4|x(t)|^4 + a_5|x(t)|^5 \quad (32)$$
$$\theta_{PA}(t) = b_0 + b_1|x(t)| + b_2|x(t)|^2 + b_3|x(t)|^2 \quad (33)$$
$$PA(t) = f_{PA}\big(A(t)\big)\cos(\omega_c t + \psi(t) + \theta_{PA}\big(A(t)\big) \quad (34)$$

LNA only suffers from AM to AM effects, AM to PM effects are negligible in LNA operation. LNA AM to AM effect is also modeled as polynomial function of input signal amplitude and is given in Eq. 35. LNA output expression is shown in Eq. 36.

$$f_{LNA}(t) = l_0 + l_1|x(t)| + l_2|x(t)|^2 + l_3|x(t)|^3 + l_4|x(t)|^4 + l_5|x(t)|^5 \quad (35)$$
$$LNA(t) = f_{LNA}(f_{PA}\big(A(t)\big)\cos(\omega_c t + \psi(t) + \theta_{PA}(t)) \quad (36)$$
$$= B(t)co\,s\big(\omega_c t + \psi(t) + \theta_{PA}(t)\big)$$

The receiver mixer model is same as the transmitter one.Mixer1 and Mixer2 outputs are given in Eq. 37 and 38 respectively. Mixer response is passed through a low pass filter to obtain the received I and Q data.

$$MX_{REC1}(t) = g\big(B(t)\big)\cos\big(\omega_c t + \psi(t) + \theta_{PA}(t)\big)\cos(\omega_c t + \phi_{MX1}(t)) \quad (37)$$

$$MX_{REC2}(t) = g\big(B(t)\big)\cos\big(\omega_c t + \psi(t) + \theta_{PA}(t)\big)\sin(\omega_c t + \phi_{MX2}(t)) \quad (38)$$

3.3 Learning Assisted Validation

There has been immense effort in silicon industry towards seamless integration of heterogeneous functionality on the same die. Technology scaling, and sophisticated packaging techniques are driving dense integration. However, high levels of device integration decreases the controllability and observability of the internal nodes of a design making design debug a difficult task. Further single design components may have multiple modes of operation that make pre-silicon verification very computation intensive and difficult. There has been an effort to use post silicon validation to capture not only electrical bugs, but also design bugs which escape pre silicon simulation based verification checks [67, 68]. There has been research on transaction based bug diagnosis for digital functionality validation [69, 70], but RF/analog circuit validation (root cause analysis of the bug, bug localization, debug techniques) [7, 71, 72] is largely unsolved. Challenges in post silicon validation of RF/Analog circuits come from:

1. Limited model information (incorrect model parameter values).
2. Limited observations (limited silicon measurement data)
3. Limited sample size of test data measurements.

Model incompleteness as described above can be of two types:

(a) All the physical aspects of the circuit is captured in the model, but the parameter values describing the physical phenomenon are not known.

(b) Some of the rare physical aspects of the circuit are not captured in the model.

In previous section [7] we have demonstrated the validation methodology by model tuning for above mentioned case (a) scenario. A model update procedure based on circuit knowledge (already known physical sources of anomaly in the circuit) is also explained in [7, 71]. An automated model building procedure for post silicon validation of Analog Circuits is described in [73]. Some physical aspects are difficult to incorporate in pre silicon model (i.e. ground bounce, cross talk, couplings etc.) and they may be rare in occurrence. To investigate into model incompleteness, these rare occurring events need to be instantiated in order to observe the model inadequacy. As the test stimuli is predicated on models, these stimuli are not good enough to show the model and DUT behavioral anomaly. In [72] the authors developed a stimuli generation technique (RAVAGE) which can overcome this problem, as it is not predicated solely on model. Validating analog circuit with small sample of measured die result has been demonstrated in [74].

In this section we will address the issue of model incompleteness in post silicon validation, and devise an intelligent validation methodology that can overcome the model incompleteness issue even if some of the physical aspects are missing from the model. This validation technique is demonstrated on two examples; a Polar Radio Transmitter and a cascaded RF transmitter chain.

The rest of the chapter is arranged as follows: In Section 3.3.1 a brief discussion of neural networks is given. Section 3.3.2 describes the Atomic Model formulation and debug

methodology. Section 3.3.3 describes the behavioral models for Polar Radio and an RF transmitter used in the experiments conducted in this research for concept validation. Simulation data is presented in Section 3.3.4 followed by conclusions in Section 3.4.

3.3.1 Non Parametric Learner

The problem of model incompleteness in post-silicon validation discussed earlier, can be mapped to a mathematical problem of non-parametric function estimation in presence of noisy data points. Two most popular non-parametric estimators are Neural Net and Kernel Estimation. In [75]the authors have demonstrated the functional estimation capability of an Artificial Neural Network (ANN) and in [76] kernel based functional estimation feasibility is demonstrated. In non-parametric data analysis, prediction and classification, Neural Net and Kernel Estimation techniques are widely used. In non-parametric data analysis (data fitting and regression) the structure or the model is not defined a priori, the model is evolved from data; where as in parametric regression the model is defined beforehand and parameter values are found from data. This non parametric data fitting regression capability of Neural Net is leveraged here [77]. Neurons (computing nodes) are the constituent elements of Neural Network. As the Figure 39 shows neurons are composed of multiplicative weight (w), additive bias (b) and a transform function (TF). A feed forward Neural Net is comprised of mainly three layers, input layers (process inputs), hidden layers (involves in data fitting for regression), output layer (process output). There may be several hidden layers in-between the input and output layers (Figure 40). In this work feed forward multilayer Neural Network is used as learner/function-estimator.

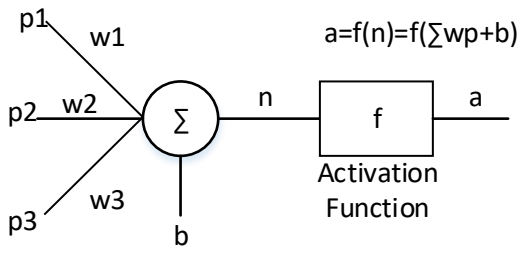


Figure 39: Neuron model

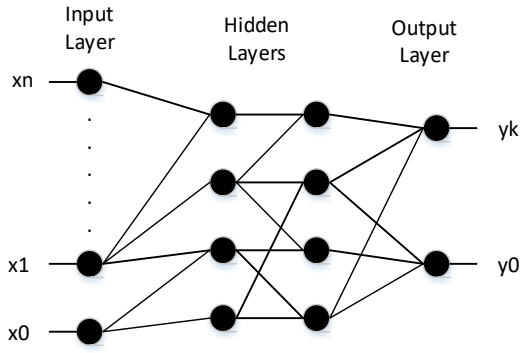


Figure 40: Neural network

3.3.2 Atomic Agent Learning

Figure 41 describes the bug localization and diagnosis strategy. This is done in two phases. In phase one the DUT and the Model is stimulated by same stimulus and their responses are captured and compared. If DUT and model responses are matched then there is no discrepancy but if it does not match, then there is a discrepancy and the goal is to localize that discrepancy to any specific sub-blocks (finding the root cause of that mismatch). To match the DUT and model responses, the first thing tried here is to modify the model parameter values as described in [5, 6]. If modifying the model parameter values of a sub-block (keeping model parameter values of the other sub-blocks at their nominal

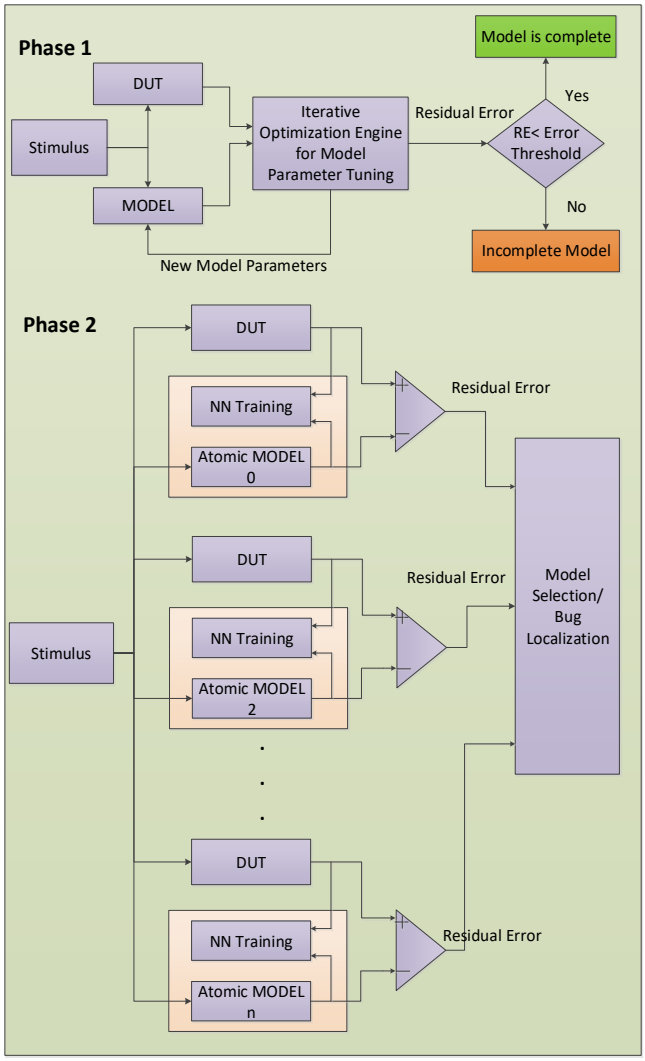


Figure 41: Bug localization algorithm

values) can match the model and DUT response, then that sub-block in DUT is inferred as the root cause of the discrepancy. The bug is then tracked down (localized) to that specific sub-block. This approach assumes single fault (only one sub-block can behave anomalously in the system) excitation. This approach is good only when model is complete, i.e. model contains all the effects that the DUT can experience. If the model is incomplete, i.e. the model does not contain the effect the DUT is experiencing, then model parameter

tuning is of no help to track down the root cause of the discrepancy between model and DUT responses.

To solve this model incompleteness issue, the concept of *atomic model learner (phase two)* is introduced here. When model parameter tuning fails to describe the discrepancy, foreign atoms are introduced in the model (as shown in Figure 42, Figure 43 and Figure 44) to match model and DUT responses. These atoms are multilayer feed forward Neural Networks. In this chapter it is numerically shown that the position of these atoms in the system can diagnose the discrepancy between model and DUT responses. The underlying concept of atoms works on the following principles

1) Atoms are basically compensating networks. When an atom is introduced in the system and it is tried to optimize the atom to match the DUT and model responses, the atom tries its best to compensate for error (between model and DUT responses) by changing its response.

2) An atom will be successful to do that if it is introduced exactly where the discrepancy comes from i.e. the source of the bug.

3) Away the atom goes from the actual source of discrepancy, its ability to match the model and DUT responses decreases.

4) An error is always best possible to compensate at its source. It is more difficult to compensate when it propagates through different transfer function into various branches.

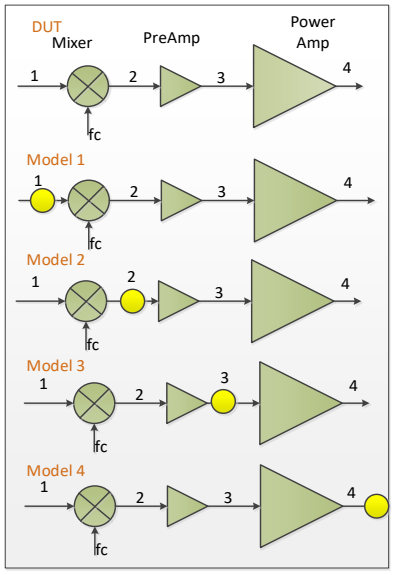


Figure 42: RF chain DUT and corresponding models

Figure 42 is showing DUT (RF transmitter) and corresponding atomic models for validation. In this DUT we have access only at the input (signal input of mixer) and at the output node (PA output). In this thesis chapter single fault (here fault means any anomalous behavior from expected behavior) assumption is made, i.e. only one sub-module can go faulty in a system. Here the goal is to identify the faulty sub-module by analyzing the output signatures from model and DUT. The above circuit have 4 nodes (as shown in the Figure 31) so 4 possible atom insertion is possible in it and hence 4 possible atomic models are generated. If only mixer model is incomplete, atom inserted at node 1 can compensate it properly, compare to atoms inserted at other nodes, as node 1 is nearest to the source of the anomaly. Similarly Power Amplifier behavior anomaly can be best compensated by atom inserted at node 3. In this example all the sub-blocks (Mixer, Pre-Amplifier and Power Amplifier) show nonlinear behavior. If all the sub-blocks behave linearly then it is mathematically impossible to distinguish among them only looking at the input and output behavior. Their nonlinear behavior entails this atom insertion based diagnosis. Figure 30

(Phase 2) is depicting the atom based diagnosis technique when model parameter tuning failed. Same stimulus is applied to DUT and all the models (inserted atoms at different places) and their response is captured. Atomic Neural networks are trained to match the model and DUT response. The best atomic model is chosen, and from the atom position the root cause is diagnosed.

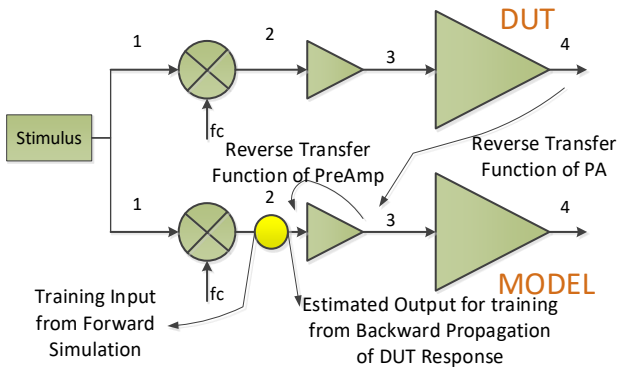


Figure 43: Constructing input and output of the neural network for supervised learning

It is to be noted that the learning technique used here is a supervised learning technique. For supervised learning input and corresponding expected outputs are required. When an atom is inserted at a node in the signal flow graph of a circuit, input to the atom (Neural Network) is obtained from forward simulating the circuit (Input to the circuit and model parameters are known). Corresponding output of the neural net is estimated by backward propagation of the expected output (Output of the DUT), to the output of the atom (Figure 43). For backward propagation reverse transfer function of the circuit blocks are used. Here it is assumed that the reverse transfer function of a circuit block exists.

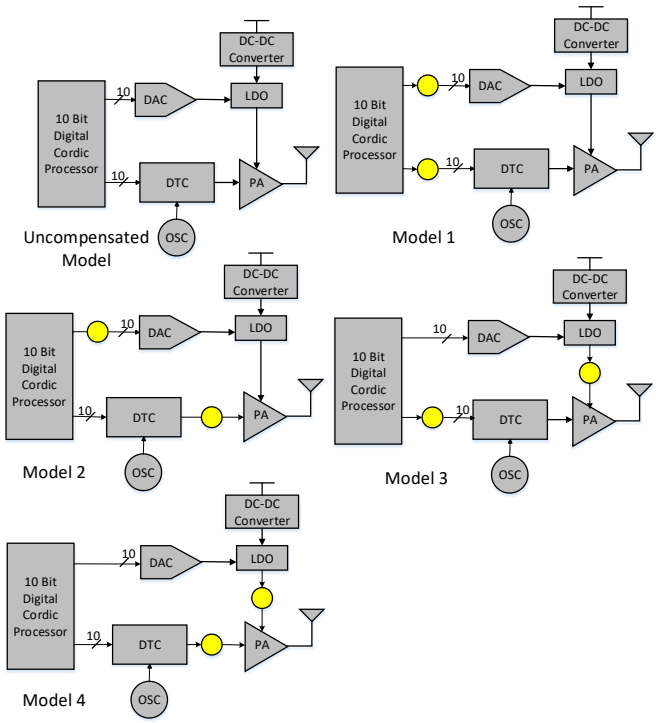


Figure 44: Polar radio DUT and corresponding models

The same best model based diagnosis for root-cause finding is applied to Polar Radio example also. Figure 44 is showing an example where CORDIC processor is faulty and four different models are used to diagnose the root cause. In simulation section we have shown that the model 1 is found to be the best model for this fault and CORDIC processor is rightly diagnosed as faulty.

3.3.3 Circuit Models

3.3.3.1 Polar Radio Model

Polar Radio Transmitter blocks are shown in Figure 45. It consists of the following sub blocks:

1. *Symbol Mapper (maps the incoming digital bits to corresponding symbols)*
2. *10 bit digital CORDIC processor.(from I and Q bits generate digital amplitude and digital phase bits)*
3. *DAC (convert digital amplitude to analog amplitude)*
4. *DC-DC converter and LDO (Control the Power Supply of the PA)*
5. *Digital to time Converter (Phase shift in accordance with the phase bits)*
6. *PA (Amplify and modulate the signal to be transmitted)*

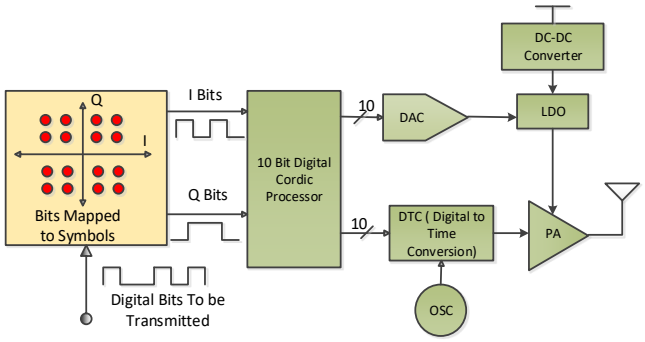


Figure 45: Polar radio transmitter

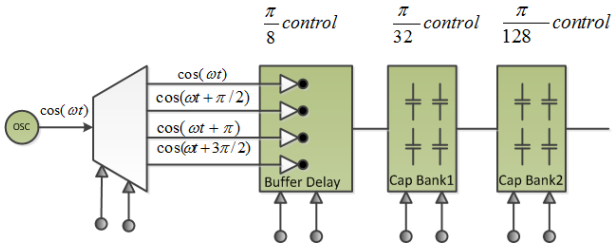


Figure 46: DTC implementation with capacitor banks

DNL, INL and SFDR non idealities are incorporated in DAC model. PA model incorporates AM to AM (input power supply voltage amplitude to output envelope amplitude) and AM to PM (input power supply voltage amplitude to output phase) non idealities. DC-DC converter & LDO block (shown in Figure 47) has finite Bandwidth. DTC is implemented with banks of capacitors (shown in Figure 46). Variation of these

capacitors with temperature, process and aging is incorporated in the model. Phase mismatch between amplitude path and phase path is also taken care of in the model.

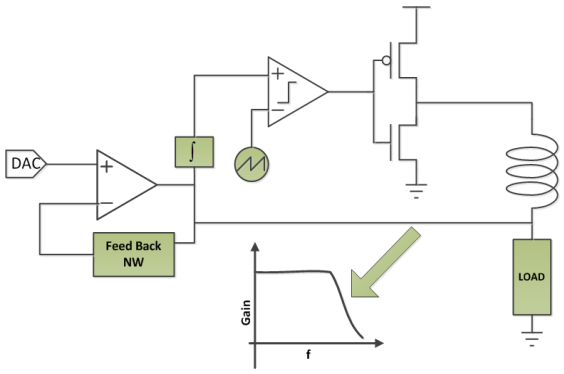


Figure 47: DC-DC converter & LDO

A brief, succinct model description is presented here. Detailed description of the analog blocks can be found in [78-81]. CORDIC processor implementation can be found in [82].

3.3.3.2 RF chain Model

RF chain DUT is shown in Fig. 4. It contains a Mixer, Pre Amplifier and a Power Amplifier in cascade. AM to AM effect, DC offset due to carrier feed through are considered in Mixer model. AM to AM in Pre Amplifier is taken care of in its model. AM to AM and AM to PM effects are captured in Power amplifier model. All AM to AM and AM to PM non linearity effects are modeled as polynomial function of input amplitude. Eq. 39, 40 and 41 describe PA, Mixer, Pre-amplifier behavior respectively. Eq. 42 and 43 describe AM to AM and AM to PM nonlinearity model respectively.

$$Y_{PA}(t) = A(t)\,cos\big(\omega t + \phi(t)\big) \quad (39)$$

$$Y_{Mixer}(t) = A(t)\,cos(\omega_c t)cos(\omega_m t) + DC\_Offset(t) \quad (40)$$

$$Y_{PreAmp}(t) = a_0 + a_1|x(t)| + a_2|x^2(t)| + \cdots . a_5|x^5(t)| \quad (41)$$

$$A(t) = a_0 + a_1|x(t)| + a_2|x^2(t)| + \cdots . a_5|x^5(t)| \quad (42)$$
$$\phi(t) = b_0 + b_1|x(t)| + b_2|x^2(t)| + \cdots . b|x^5(t)| \quad (43)$$

3.3.4 Simulation Results

3.3.4.1 Polar Radio

Figure 48 shows fault diagnosis and bug localization in a Polar Radio Transmitter system. Here x-axis represents model number and y-axis represents residual error after model tuning and compensation. In this case the bug is introduced in CORDIC processor. In CORDIC processor RTL some nodes are made to stuck at a certain binary value (stuck at 0 and stuck at 1 faults are introduced) independent of the gate inputs, driving those nodes. This experiment is repeated for 10 different random fault sets. In each experiment 15 random nodes are selected and their stuck at values are also randomly generated. This Cordic RTL (with Fault) along with the other sub-blocks (non faulty) constitute the DUT for this experiment. Four different atomic models (shown in Figure 44) have been used to diagnose this fault. For all ten fault sets the minimum error (Error between DUT and model response) is obtained for model 1. The two atoms in model 1 are nearest to the fault site than the atoms in other models.

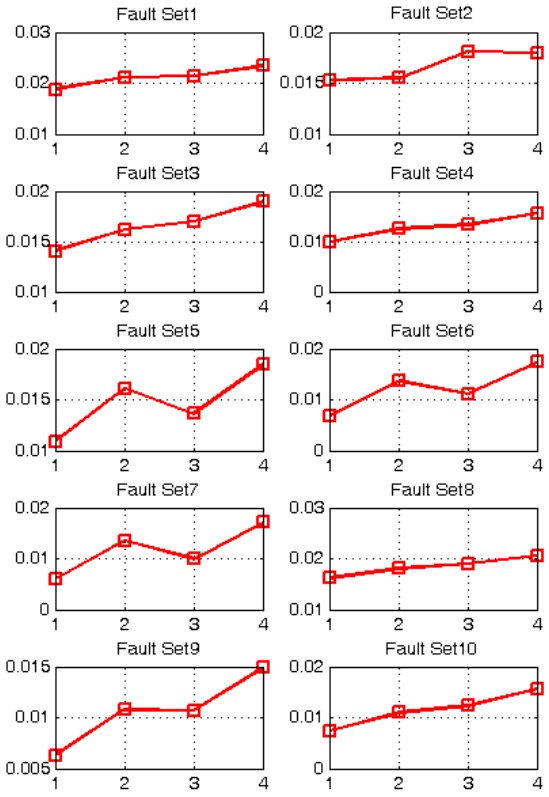


Figure 48: Polar radio fault diagnosis (cordic processor faulty)

Table 9: Residual error

Experiment Number	Logarithm of Residual Error		
	DAC	PA	CORDIC Processor
1	-21.9043	-11.7049	-6.7326
2	-22.3584	-7.1094	-6.3264
3	-22.2561	-14.0340	-6.3462
4	-22.3765	-12.1247	-6.1759
5	-22.3437	-13.0174	-6.1924
6	-22.0013	-10.0827	-5.8924
7	-22.5513	-12.5676	-6.4661
8	-22.1433	-11.2577	-6.0062
9	-22.1553	-15.2747	-6.3823
10	-22.3880	-9.3034	-6.7486

Instead of using atomic learners in between two nodes for fault diagnosis, the approach of [7, 71] i.e. in place of a sub-module an atomic learner can be placed. Here instead of tuning the parameters of the sub-block model, the learner would strive to develop a local model (local w.r.t the input stimulus) of the sub-block. The same diagnosis argument is valid here too. Here is presented an example where the current source of the DAC for 10^{th} bit was not turned on due to a design bug (logic bug in the digital controller of the DAC). First the bug is localized to the DAC block. Then the transfer function estimated from the Neural Net is analyzed to predict exact bit position of the bug. In this experiment three different atomic models are used. In 1^{st} model DAC is replaced with an atom, and similarly in 2^{nd} and 3^{rd} model PA and CORDIC processor are replaced with atoms respectively. Table 9 is showing the logarithm of the residual errors for the above mentioned three models. From this table it is clear that the anomalous behavior is due to the DAC (replacing DAC with an atom produces best matching between DUT and model responses). After localizing the source to the DAC, the transfer function of the atom that replaces DAC are looked into (Figure 49). There is a discontinuity at code 512, the transfer

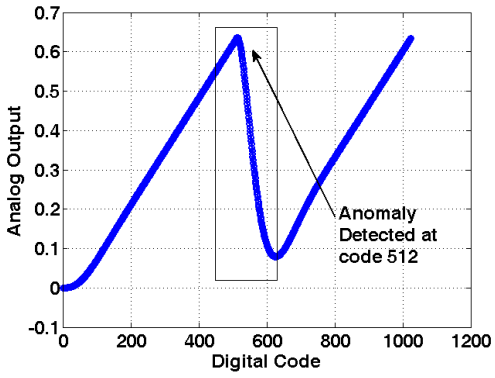


Figure 49: DAC transfer function estimated from DAC atom

function is not monotonic. From this behavior of DAC transfer function, 10th bit of the DAC is considered the source of anomaly in the system.

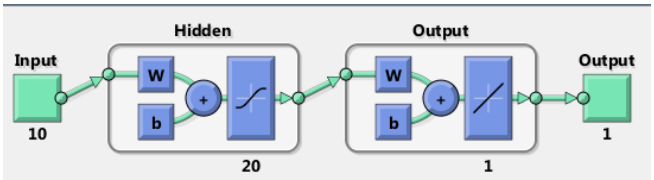


Figure 50: Neural network used for DAC

In all the Polar Radio experiments Feed forward neural net with 20 hidden layer is used (as shown in Figure 50) and for training the network Levenberg-Marquardt back propagation algorithm (Matlab trainlm [83]) is used.

3.3.4.2 RF chain

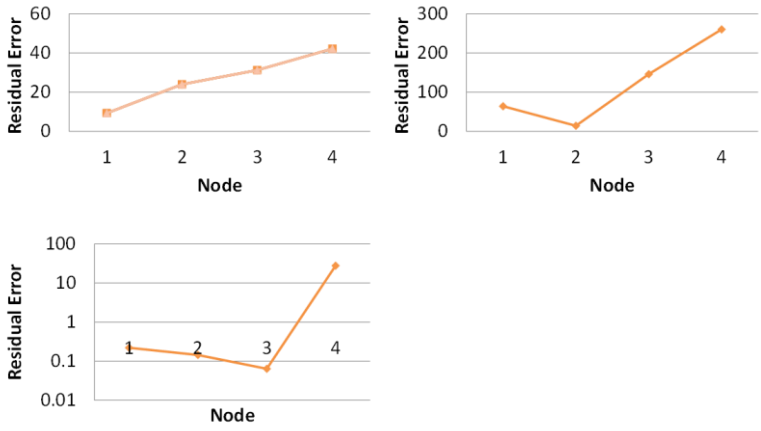


Figure 51: Root cause diagnosis of RF chain

Figure 51 is showing the simulation result of fault diagnosis in a RF chain. Three fault case is considered here; case 1 (Mixer is faulty), case 2 (Pre amplifier is faulty) and case 3 (Power amplifier is faulty).In all the experimental cases considered, it is observed that as the atomic learner goes away from the fault site, less effective it is in matching the

DUT and model response. This put forward a strong basis for fault isolation in a cascaded system. All these experiments numerically avouch the claim of this thesis chapter(i.e. position of the atoms can diagnose the root cause of anomalies in post silicon behavior of DUT and its corresponding model).

3.4 Conclusions

Model based validation work presents a novel heuristic model generation and model adequacy checking methodology for synthesis of DUT model. If the model is adequate, a unique model parameter computation based discrepancy isolation technique for RF/ analog circuits and systems is proposed here for post silicon validation. An OFDM transceiver is used to show the feasibility of this approach. If the circuit is modeled explicitly i.e. circuit's output response is explicitly modeled as circuit parameters (closed form equations are known), only few model simulations and iterative nonlinear optimizations are sufficient to isolate/localize the discrepancy to specific design module.

At initial stage of post silicon validation, complete model is rarely available. We have addressed the model incompleteness issue in RF/Analog circuit validation, in learning assisted validation approach. A self-learning methodology for bug localization even in presence of incomplete model is demonstrated here. A Polar Radio transmitter and a RF chain transmitter are used as test vehicle to attest the viability of the learning method in post silicon validation. This learning methodology is also able to remove the assumption (functional form of the non-idealities in the model) the authors made in [7] at the cost of computation (Training Neural Network) .

CHAPTER 4. BISCC: EFFICIENT PRE THROUGH POST SILICON VALIDATION OF MIXED-SIGNAL/RF SYSTEMS USING BUILT IN STATE CONSISTENCY CHECKING

4.1 Introduction

Aggressive scaling of device technology has enabled massive integration in today's integrated circuits. Though large scale heterogeneous integration has helped in incorporating newer features and functionality in the same die area, state of the art SoC's pose daunting test and validation challenges. Pre-silicon, a key challenge is to identify design bugs rapidly without the need to rely on human generated assertions for design validation. Post-silicon, the low controllability and observability of internal circuit nodes in modern SoCs poses a significant challenge. Besides logical bugs, difficult-to-simulate electrical bugs that cause silicon malfunction pose major validation challenges as well. Such electrical bugs due to signal crosstalk and power supply-ground bounce for example, occur under rare input stimulus conditions and are difficult to detect and diagnose. Clearly, new pre and post-silicon design validation methods are urgently needed that are completely automated and do not require the use of manually generated assertion based design checking procedures. They must provide high design bug coverage, allow detection of design bugs with low latency and facilitate diagnosis of design bugs down to subcircuits of a large design for rapid manual analysis and redesign.

4.2 Prior Work

Pre-silicon verification methodologies for analog/mixed signal systems can be broadly classified into three groups: i) based on equivalence checking **[84]** ii) based on model checking **[85]** and iii) based on specification testing using SPICE simulation. SPICE simulation is computationally prohibitive for AMS system verification. The main drawbacks of the state of the art validation techniques employing i) and ii) are: (a) assertions for checking design correctness are hand crafted and require input from experienced analog designers, (b) only very simple properties and specifications of AMS circuits and systems can be handled. Fault isolation and diagnosis are not addressed adequately by state of the art AMS verification methodologies and is largely solved by manual simulation. Further pre-silicon verification techniques discussed above are not readily amenable to mixed-signal/RF post silicon validation.

With regard to post-silicon validation, scan chains are popular in digital design for providing state observability and controllability that aid circuit debugging. . Although the analog scan standard IEEE 1149.1/1149.4 **[86, 87]** is in practice for board level debugging, it has not found widespread application in circuit level testing of analog/RF IP blocks in SoCs. Current based analog scan chain was proposed in **[88, 89]** by Soma et.al. Popular analog scan methods relay on voltage to frequency conversion and voltage to delay conversion **[90]**. However, such methods are not suitable for testing AMS circuits at-speed, making high-speed AMS system testing a difficult task. In **[91]** the authors have proposed an analog DFT technique that relies on supply voltage ramp-up. The captured current signatures from various IP blocks are compared against known thresholds and multiple digital bits are generated as test response signatures. All the above scan based test methods

suffer from the inability to test AMS circuits and systems at-speed, where they are most vulnerable to design bugs.

In contrast, analytical model based analog/RF post silicon validation has been proposed in **[7, 71]** and model learning based diagnosis in **[92]**. A key issue with model driven validation is that the nature of complex design bugs is generally not known a-priori and the "difficult" electrical bugs due to coupling and ground bounce, for example, cannot be easily simulated.

With regard to test stimulus generation, there has been research on test generation driven validation in the digital space **[93, 94]**. However, there is not much parallel work in the mixed-signal/RF domain. In [95, 96]**,** the authors use *diverse programs* with the *same functionality* to detect design bugs in processor cores, uncore components and accelerators. A hardware design bug is detected if there is any inconsistency in the results obtained from the two functionally equivalent program streams. The test procedure does not make any assumption about the nature of design bugs and the extent of design bugs uncovered is limited only by the diversity of test programs deployed.

In our proposed approach, a reference model of an AMS system is used to design spectrally diverse test stimulus for the system or module under test. The analogy to the work of **[96]**, is that in the latter, program diversity is dictated by the instruction set architecture (ISA) of the processor being debugged (reference model). The diverse stimuli are designed to take the circuit under test from a known initial condition to the *same final state* (measured voltage/current values at specified circuit nodes). The final states reached by the two diverse stimuli, applied in sequence, are acquired using track-and-hold circuitry

(see Figure 1, the comparison results are scanned out using digital scan chains). By checking for consistency between the final states reached by the diverse test stimuli, design bugs are detected with low latency and high coverage (see Figure 1). As in **[96]**, no assumption is made about the nature of design bugs detected (models for hard design bugs are developed only *after* they are detected with considerable debug and bug modeling effort).

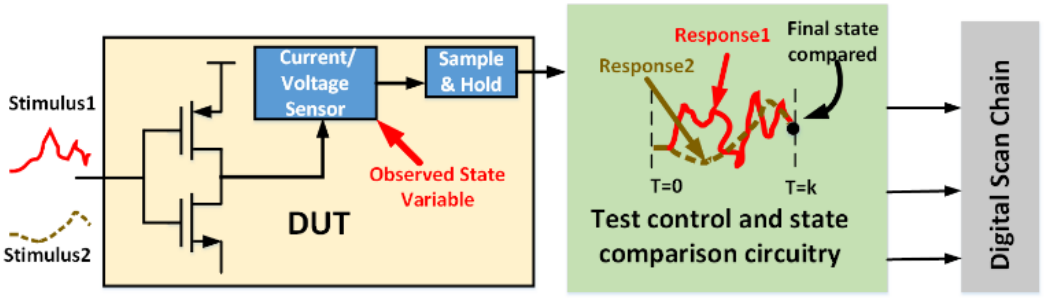


Figure 52: State consistency checking based validation of mixed-signal/RF systems

The main contributions of this work are as follows:

1) *The proposed BISCC approach for design debug is a unified methodology for pre and post-silicon validation of AMS/RF systems and can be applied seamlessly through the design process with appropriate modifications.*

2) *The design validation approach of BISCC does not rely on the use of manually generated assertions in the AMS domain (that are prone to errors) for design checking. In contrast, it can automatically check for design bugs, even bugs whose effects are unknown prior to circuit debugging.*

3) *BISCC uses short test sequences for detecting bugs that require very low test application time. Since only the final states of two diverse test sequences are*

compared, the volume of test data generated is small and in post-silicon, be easily scanned out using existing digital scan chains in SoCs. Further, the tests post-silicon, are generated using existing digital processors on-chip or using minimal amounts of compact digital stimulus generation logic with low hardware overhead.

4) *For post-silicon debug, the BISCC methodology entails the use of minimal on-chip hardware (as compared with other analog/RF DfT techniques [4-6, 12]) consisting of track-and-hold and voltage/current comparison circuitry as opposed to full ADCs with attendant signal routing and fidelity issues. Moreover, following from this simplicity, the use of multiple voltage and current test points allows localization of design bugs to specific design modules for system debug.*

The rest of the chapter is organized as follows: section 4.3 describes the underlying basic principles of analog state consistency checking based debugging. DFX infrastructure placement algorithms are discussed in section 4.4. Rapid stimulus generation algorithms are explained in section 4.5. Analog signal capturing and comparing DFT circuits are explained in section 4.6. Test vehicles used to demonstrate the feasibility of the proposed approach are described in section 4.7. Simulation results to corroborate the proposed claim are shown in section 0. Finally future possible work and conclusion are given in section 4.9.

4.3 Debugging Using Analog State Consistency Checking

4.3.1 Analog State Space Model (ASSM) Representation

A formal definition of state variable and state consistency is given in Table 10. Intrinsic as well as parasitic device/interconnect resistance, capacitance and inductance make any RF/Analog circuit a state machine. Number of possible states in such state machine is infinite as analog currents/voltages can take any values within the dynamic range. Here any branch current or node voltage can be thought of as state variable of the system.

Table 10: Formal definitions of state variable and state consistency

	Definition
State Variable (SV)	*Any observed node voltage or branch current is a state variable.* $SV \in \mathbb{R}$ *and* $R_{i1} < SV_i < R_{i2}$ *Where* (R_{i1}, R_{i2}) *is the dynamic range of the observed current/voltage corresponding to state variable i.*
State Consistency	***Temporal Consistency:*** *Two state values* SV_i^1 *and* SV_i^2 *of state variable* SV_i *will be consistent with each other if* $\left\|SV_i^1 - SV_i^2\right\| < \epsilon$ *where* ϵ *is comparator offset voltage* ***Spatial Consistency:*** *Two state values* SV_i^1 *and* SV_j^1 *of state variables* SV_i *and* SV_j *will be consistent with each other if* $\left\|SV_i^1 - SV_j^1\right\| < \epsilon$ *where* ϵ *is comparator offset voltage*

4.3.2 State Consistency Checking Approach:

Type I Test (Temporal State Consistency Checking): In our approach a piecewise linear stimulus of duration T is crafted across a time grid of spacing δ and N grid points where $T = N\delta$ (see Figure 53). At time t=T the final value of the state variable SV in response to stimulus S1 is sampled and held using a sample and hold (S/H) circuit for additional time T. Between time t=T to t=2T, a different stimulus S2 is applied to the DUT starting with the same initial condition as in S1. Final value of the state variable SV in response to S2 is acquired at t=2T using a S/H circuit. Subsequently the sampled values of SV at t=T and t=2T in response to the applied stimuli S1 and S2 respectively, are compared. If they are

consistent (see Table 10 for consistency definition) then a logic “0” is generated by the error triggering circuit (see Figure 52) else a logic “1” is generated.

The core idea behind temporal state consistency checking is to design the stimuli S1 and S2 in such a way that they are diverse (exercise the analog/RF circuit through diverse state trajectories) but result in consistent final states sampled at t=T and t=2T. This leads to the hypothesis that an arbitrary design bug or fault is unlikely to affect the state trajectories of SV in response to S1 and S2 identically and thereby results in inconsistent final state values.

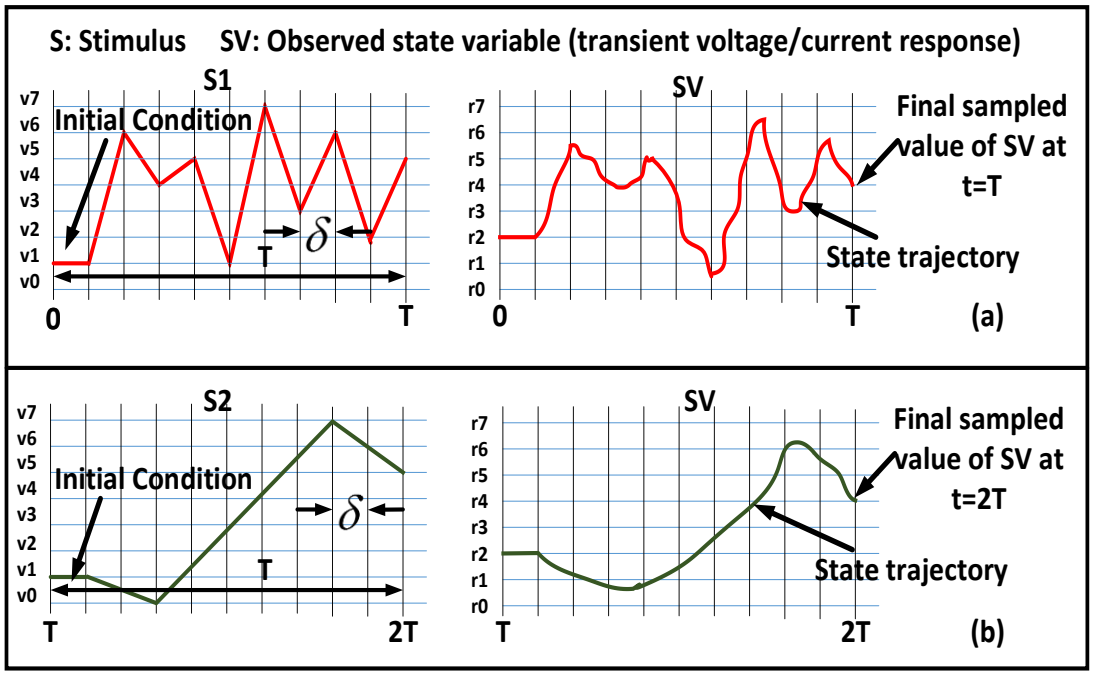


Figure 53: Temporal state consistency

Type II Test (Spatial State Consistency Checking): Type II tests are designed to check consistency across state variables at same sampling instant as opposed to type I test where state consistency of a state variable is checked between two different sampling instants. A

stimulus will be generated so that the two observed state variables are consistent (see consistency definition in Table 10) for nominal circuit at sampling instant t=T (see Figure 54). It is to be noted that two observed state variable pair may be having different dynamic ranges, so proper level shifting and gain compression are required before comparison. There are some pathological cases such as gain compression, DC offset where the faulty circuit may show state consistency under type I stimuli test. To catch these faults we introduce type II test, where state consistency across state variables are checked.

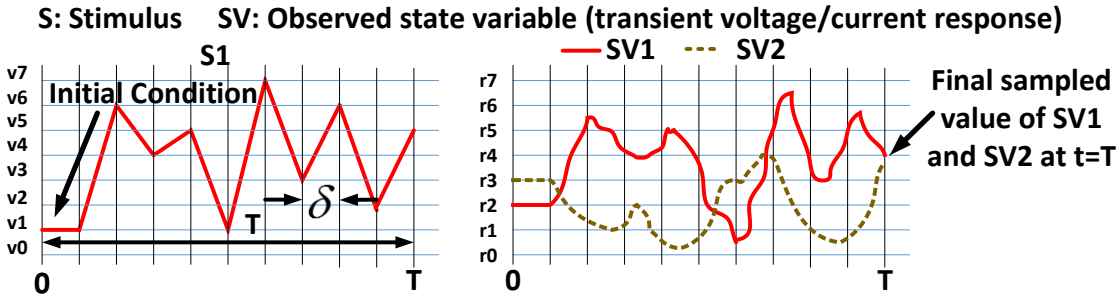


Figure 54: Spatial consistency checking

4.4 DFX Infrastructure Placement

In previous section we have discussed state consistency checking of state variables. How many state variables are needed to diagnose a system, and which ones to be cherry picked will be discussed in this section. Every state variable is observable in pre-silicon stage. DFX structures are to be placed in design to make selected state variables observable in post-silicon stage. In simulation we select all possible non-intrusive DFX sensor positions in the system, and for a long random stimulus collect sensor data from respective sensors. We will keep those sensors that are volatile based on a threshold volatility. Higher the volatility, richer the sensor is with information about the system. As the dynamic ranges

of the all the sensors are not same, we have used a scale free volatility measure given in Eq. 44.

$$volatility = standard\ deviation/mean \tag{44}$$

Table 11: State variable selection algorithm

Given: State Variable Set $SVS = \{SV_1, SV_2, \dots \dots SV_m\}$
Objective: Find State Variable Set SVS1 for type I test and SVS2 for type II test
Step 1: Take a long random stimulus and stimulate the system to collect the system response. All state variable values corresponding to the applied random stimulus are acquired.
Step 2: SVS1={}
For i=1 to m
    If volatility(SV_i) > Threshold Volatility
        $SVS1 = SVS1 \cup SV_i$
Step 3: $k = |SVS1|$ (cardinality of set SVS1) SVS2={}
For i=1 to k
    For j=1 to k
        If $i \neq j$
            Max_crosscorr=max(xcorr(SVS1(i),SVS1(j)))
            If Max_crosscorrelation > ThresholdCrossCorr
                $SVS2 = SVS2 \cup \{(SVS1(i), SVS1(j))\}$

Let's assume that the type I test set be SVS1 and is shown in equation 45.

$$SVS1 = \{SV_1, SV_2 \dots SV_k\} \tag{45}$$

For type II test we check state consistency between a pair of state variables at same sampling instant. From type I test set, we constitute pairwise variable set (as shown in Eq. 46) and compare cross correlation between every pair.

$$\left(SV_i, SV_j\right)\ s.t\ i \neq j\ and\ i,j = 1(1)k \tag{46}$$

In a circuit the observed state variable pair may be phase shifted. For cross correlation enumeration matlab function *"xcorr"* is used. *xcorr(x1, x2)* finds cross

correlation between *x1* and *x2* at every shifted position of *x1* and *x2*. Maximum of all shifted cross correlation serve as a metric to pick sate variable pair. The formal state variable selection algorithm is shown in Table 11.

4.5 Automatic Test Pattern Generation

In the section, we discuss generation of BISCC piecewise linear (PWL) test stimulus. The test points are stored in a small memory (kilo bytes) and the stimulus is produced using a built-in DAC. Alternatively, seeded LFSRs can also be used, where the seeds are optimized for stimulus diversity (the pulse train output of the LFSR is directly filtered and applied to the CUT).

4.5.1 Stimuli Generation for Type I Test:

For type I test we need two diverse stimuli which will take the circuit to the same state by two different state trajectories. First two stimuli (S1 & S2) of length l are randomly generated (as shown in Eq. 47).

$$S1 = [v_{11}, v_{12}, \dots\dots\dots v_{1l}]^T \quad S2 = [v_{21}, v_{22}, \dots\dots\dots v_{2l}]^T \tag{47}$$

Their entropies and dissimilarities are checked. Standard deviation is used for entropy checking and distance correlation is used for dissimilarity checking. If $(X_i, Y_i)\ \ i = 1 \dots . n$ are samples of two vectors (X, Y) then distance correlation of the two vectors is given by Eq. 48.

$$dcor(X,Y) = \frac{dcov(X,Y)}{\sqrt{dvar(X) * dvar(Y)}} \tag{48}$$

Detailed derivation of Eq. 48 can be found in [34]. Entropy checking ensures that the signals are varying enough while the dissimilarity checking ensures that the two stimuli are different. The system is simulated with stimuli S1 and S2, and corresponding responses R1 and R2 are captured. Stimuli generation objective is to match the end responses R1(l) and R2(l) as far as possible . S1 is kept fixed, and S2 is modified in steps from rear end (S2(l) , S2(l-1)…up to S2(l-m)). S2(l) value is replaced with all possible new values and corresponding stimuli set $S2_{new}$ is given in Eq. 49. Corresponding to all the stimuli $S2_{new,}$ the system is simulated and response set $R2_{new}$ is obtained.

$$S2_{New} = \{\, [S2(1)_{old}\,, S2(2)_{old}, \ldots\ldots S2(l-1)_{old}, S2^{1}(l)]^{T}, \quad (49)$$
$$[S2(1)_{old}\,, S2(2)_{old}, \ldots\ldots S2(l-1)_{old}, S2^{2}(l)]^{T},$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots,$$
$$[S2(1)_{old}\,, S2(2)_{old}, \ldots\ldots S2(l-1)_{old}, S2^{q}(l)]^{T}\}$$

$$R2_{new} = \{\; R2^{1}, R2^{2}, \ldots\ldots R2^{q} \;\} \quad (50)$$

Stimulus S2j is selected such that abs(R1(l)-R2j(l)) is minimized. The above process is repeated for S2(l-1), S2(l-3) and so on until R2(l) reaches R1(l) within acceptable accuracy.

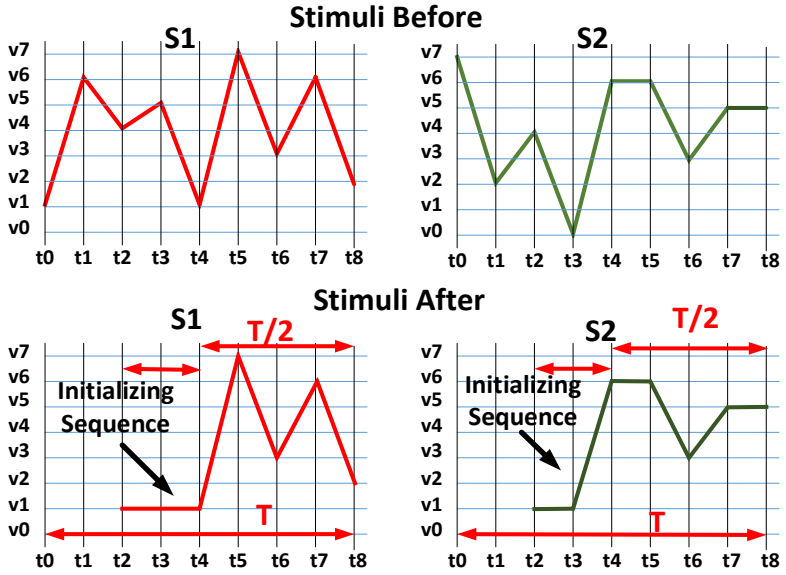


Figure 55: Stimuli length optimization

Here we have explained the algorithm for all possible quantized s2(l) values for ease of understanding. In reality a binary search is used in this work. The stimuli generation process is formally defined in Table 12.

Table 12: Type I (temporal) stimuli generation algorithm

Given: Stimuli duration (T), sampling rate (R), stimuli dynamic range (DR), Entropy Threshold, Dissimilarity Threshold
Stimuli length (l) = T/R
Quantize the dynamic range DR into q number of levels
$V = [v_1, v_2, \ldots \ldots . v_q]^T$
Step 1: Randomly sample values from V and generate two stimuli S1 and S2 of length l (as shown in equation 47).
Step 2: Perform the following checks
(i) $\sigma_{S1} > Entropy\ Threshold$
(ii) $\sigma_{S2} > Entropy\ Threshold$
(iii) $dcorr(S1, S2) < Dissimilarity\ Threshold$
If all the checks are passed go to step 3, else revert back to step 1
Step 3: Simulate the system with S1 and S2 and system responses R1 and R2 are captured.
If abs(R1(l) – R2(l)) < Comparator Offset
then end the program
Else go to step 4
Step 4: keep S1 fixed, do a binary search on S2 (l-1) to S2 (l) transition so that *abs(R1(l) – R2(l))* is minimized.
Step 5: (i) Repeat the step 4 for S2 (l-2) to S2 (l-1) transition
(ii) Repeat the step 4 for S2 (l-3) to S2 (l-2) transition
……
(m) Repeat the step 4 for S2 (l-m+1) to S2 (l-m) transition
Step 6: If abs(R1(l) – R2(l)) < Comparator Offset
Then accept the stimuli S1 & S2 pair
Else go to step 1

4.5.2 Stimuli Generation for Type II Test:

A similar approach has been employed for type II test. Here stimulus S1 is randomly generated and two responses (state variables) R1 and R2 are captured. For spatial comparison the responses R1 and R2 need to be level shifted and compressed/expanded.

We assume R1 and R2 are responses after proper level shifting and gain adjustment. S1 is modified in steps from rear end (S1(l) , S1(l-1)…up to S1(l-m)) following the algorithm shown in Table 12.

4.5.3 Stimuli Set Formation:

While generating stimuli for type I and II tests, it is made sure that the two responses match within the acceptable accuracy at the sampling instant. Two responses can match at any voltage/current level within the dynamic range of it. While creating stimuli set, a fair representation from every corner of the dynamic range is ensured.

4.5.4 Stimuli Length Optimization:

In order to reduce the test time (applicable for both pre and post silicon) and memory requirement for on chip BIST stimuli storing (post-silicon) generated stimuli are further compacted. Let us assume that the stimuli set SS_1 and SS_2 each of cardinality N (given in equation 51 and 52) are formed by following the algorithms described before. From SS_1 and SS_2 , new stimuli set SS_1^* and SS_2^* are obtained by cutting every stimulus length to half from rear end, and append an initializing sequence before the stimulus. The process is shown in Figure 55.

. For new stimuli set SS_1^* and SS_2^*, new response set RS_1^* and RS_2^* are obtained by stimulating the system with new stimuli set. For every new stimulus the corresponding response is checked if it obeys the previously checked objectives (step 2, Table 12). If more than 90% of the stimuli obey the objective, we accept the new time duration. If the condition is not satisfied we cut it down by a quarter and repeat the process.

$$Type\ I\ Stimuli\ Set\ SS_1 : \{ (S1,S2)^1, (S1,S2)^2, \ldots\ldots \quad (S1,S2)^N \} \tag{51}$$
$$Type\ II\ Stimuli\ Set\ SS_2 : \{ S3^1, S3^2 \ldots\ldots \quad S3^N \} \tag{52}$$
$$Type\ I\ Response\ Set\ RS_1 : \{ (R1,R2)^1, (R1,R2)^2, \ldots\ldots \quad (R1,R2)^N \} \tag{53}$$
$$Type\ II\ Response\ Set\ RS_2 : \{ (R3,R4)^1, (R3,R4)^2, \ldots\ldots \quad (R3,R4)^N \} \tag{54}$$

4.6 Debug Hardware

In post-silicon validation, internal current/voltages are accessed by signal capturing circuits. In RF transceiver envelop detector is used to capture low frequency signature from modulated voltage signal. For supply current sensing a small resistance R_{sense} is used to convert supply current to voltage (see Figure 56b). Further amplification and low pass filtering are done by the op-amp and low pass filter respectively.

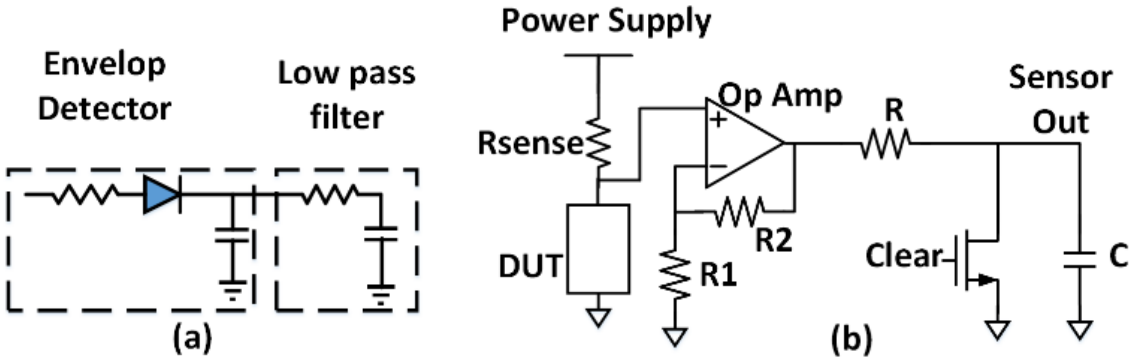


Figure 56: (a) Low frequency voltage signal capturing circuit for RF receiver (b) Supply current sensor

Two types of error trigger architectures are shown in Figure 57. Temporal architecture is used to compare between previously sampled value and present sampled value. A sample & hold circuit is used to hold the previously sampled value. For spatial comparison as the comparing signals are coming from two different circuit nodes, no such holding is required.

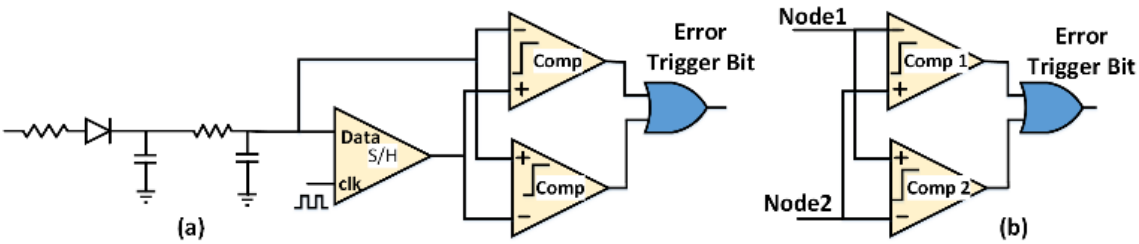


Figure 57: (a) Temporal error trigger DFT architecture (b) Spatial error trigger DFT architecture

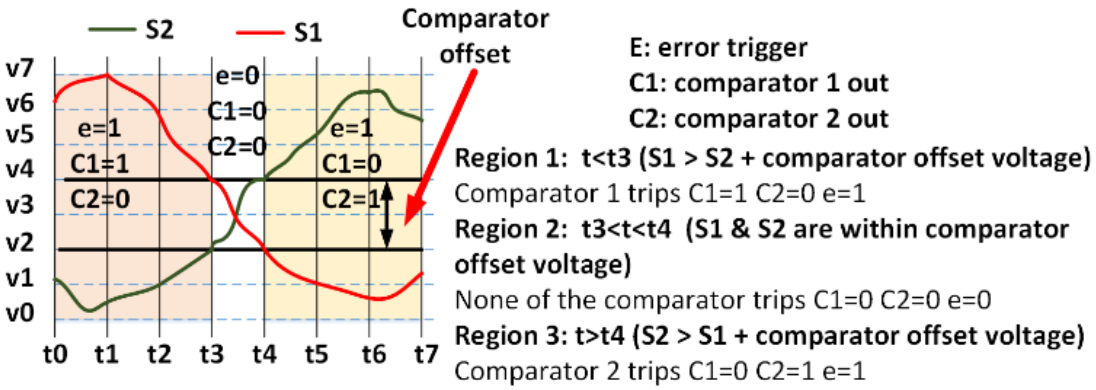


Figure 58: Error trigger operation

The error triggering mechanism is shown in Figure 58.

4.7 Test Vehicles Used

In this work a RF receiver and a sigma delta ADC are used as test vehicles to corroborate the efficacy of the proposed state consistency checking based validation methodology for mixed signal/RF systems. While the RF quadrature receiver system is designed in 130nm IBM process, the Sigma Delta ADC is designed in 45nm predictive transistor model from NCSU. The quadrature RF receiver system (as shown in Figure 60) is consisting of LNA, power splitter and two RF demodulating mixers. Transistor level

circuit design for LNA and mixers are shown in Figure 59 (a) and (b) respectively. RF and analog circuit components are design in cadence spectre, while the ADC and base band processing is done in matlab in order to simulate the receiver system. The LO frequency of the designed receiver is 2.4GHz and the in phase and quadrature phase data rate is 1 MHz.

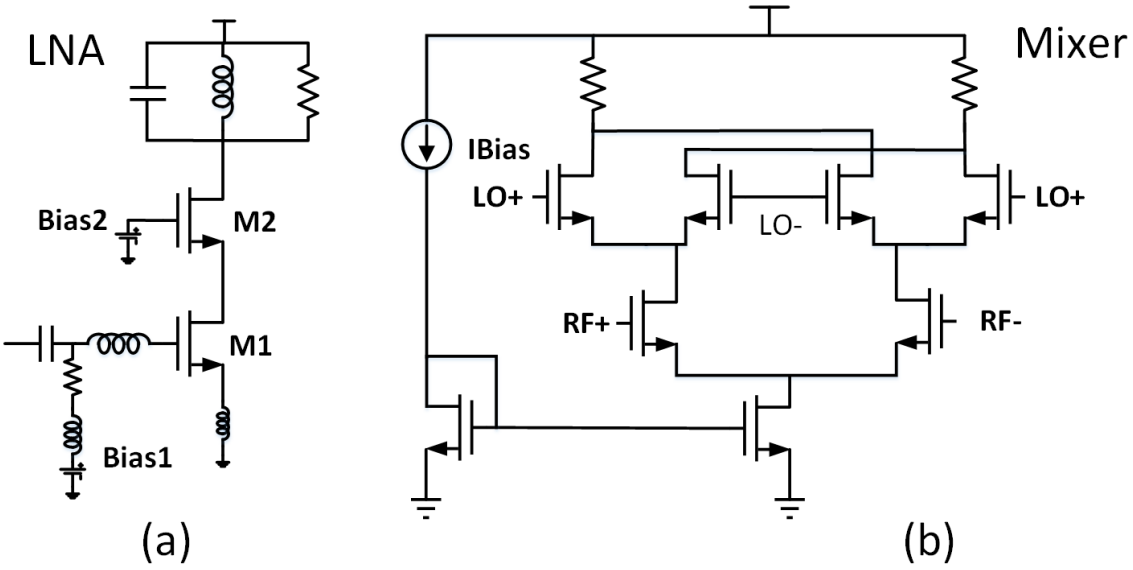


Figure 59: (a) Cascode LNA (b) Gilbert cell mixer

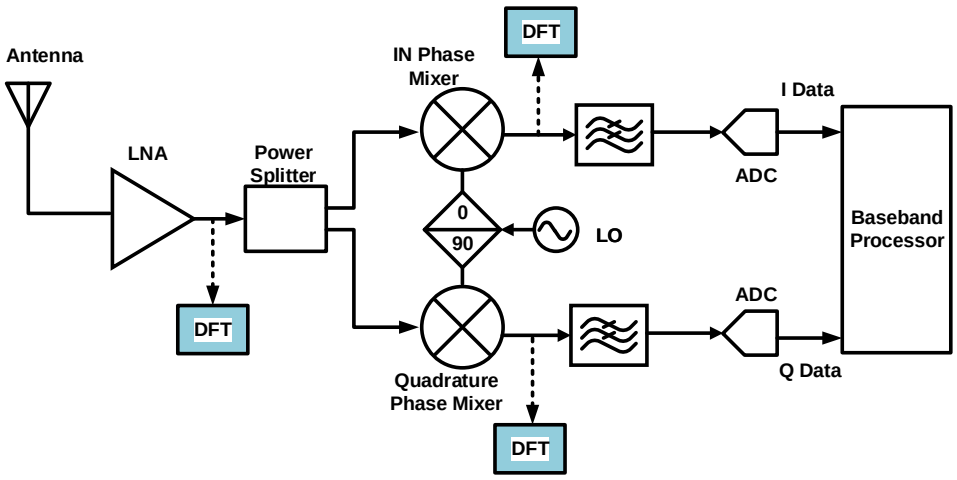


Figure 60: RF receiver

The second example, Sigma Delta ADC is shown in Figure 61. Here all the blocks, sample and hold, opamps, comparator, D Flip-Flop are designed in cadence spectre. Sampling clock and over sample clock frequencies are 1 MHz and 1 GHz respectively (over sampling ratio of 100). The nominal performance parameters are shown in Table 13.

Table 13: Nominal sigma delta ADC specifications

SFDR	THD	ENOB
30.44 dB	-10.31 dBc	4.76

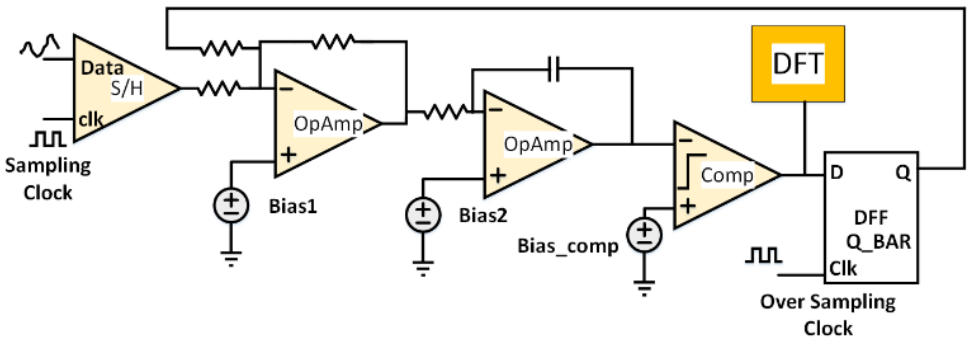


Figure 61: 1st Order sigma delta ADC

4.8 Simulation Results

4.8.1 State Variable Selection

Table 14: State variable definition for RF receiver system

State Variable	Definition	Test #
State Variable 1	voltage signature captured by the envelop detector placed at the output of the LNA	Type I Test 1
State Variable 2	voltage signature captured by the envelop detector placed at the output of the In Phase Mixer	Type I Test 2
State Variable 3	voltage signature captured by the envelop detector placed at the output of the Quadrature Phase Mixer	Type I Test 3
State Variable 4	current signature captured from LNA supply current	Type I Test 4
State Variable 5	In Phase input data	X
State Variable 6	Quadrature Phase input data	X
State Variable 7	current signature captured from In Phase Mixer supply current	X
State Variable 8	current signature captured from Quadrature Phase Mixer supply current	X
State Variable pair 1	{State Variable 1, State Variable 4}	Type II Test 1
State Variable pair 2	{State Variable 2, State Variable 4}	Type II Test 2
State Variable pair 3	{State Variable 1, State Variable 2}	Type II Test 3

Observed state variables of the RF receiver system are defined in Table 14.

Table 15: Volatility of observed state variables (RF receiver system)

	Standard Deviation	Volatility	Accept/Reject
State Variable 1	0.1112	0.0541	√
State Variable 2	0.2784	2.5187	√
State Variable 3	0.2881	6.6280	√
State Variable 4	0.3671	0.1537	√
State Variable 5	0.0021	0.0014	X
State Variable 6	0.0200	0.0014	X

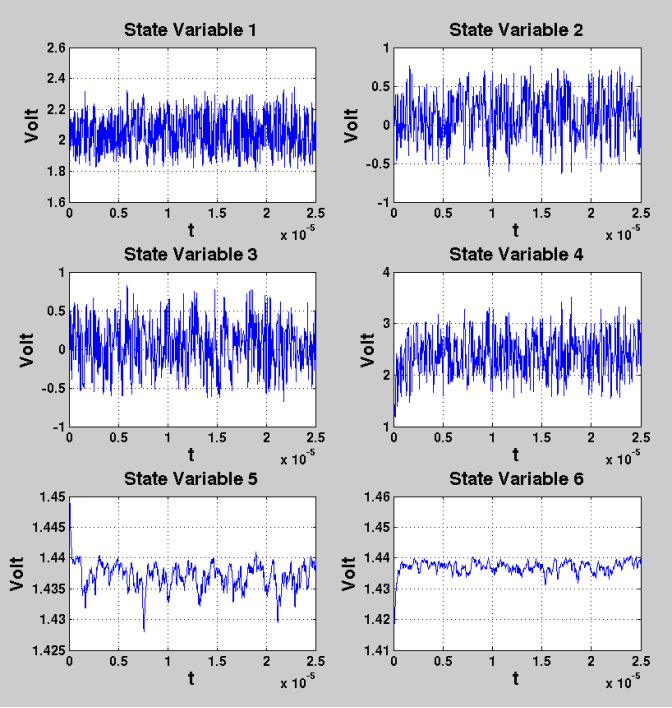


Figure 62: Observed state variables for a random input stimulus

For the RF receiver system example, 6 DFX sensors were selected and the signatures obtained for a long random stimulus are shown in Figure 62. Volatility metric of each sensor is shown in Table 15 (each signature is represented by corresponding state variable). Sensors placed at positions 1, 2,3and 4 are acceptable while the sensors 5, 6 are not providing enough information to diagnose the system. State variable 5, 6 corresponds to bias currents of In Phase and Quadrature mixer respectively. As these two mixers are bias at high DC current, their supply currents are not sensitive enough to AC input stimulus. The accepted state variables will constitute type I test set.

$$Type\ I\ Test\ Set = \{State\ Variable\ \ 1,2,3,4,7,8\} \tag{55}$$

From the type I test set (given in equation 55) we constitute pairwise state variable set and compare cross correlation between every pair. Pairwise maximum cross correlation

plot is shown in Figure 63. From Figure 63 we chose the following pairs {1, 4}, {2, 4}, {1, 2}, {4, 6}.

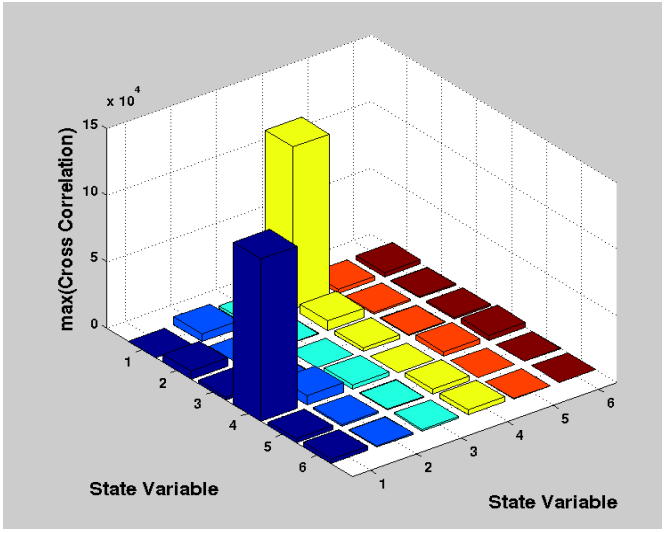


Figure 63: Pairwise maximum cross correlation among state variables

Similar experiments were done for Delta Sigma ADC and the tests defined are shown in Table 16.

Table 16: State variable definition for RF delta sigma ADC

State Variable 1	Voltage signal captured by a low pass filter placed at the output of the comparator	Type I Test 1
State Variable 2	Input sampled value at the output of the S/H circuit	X
State Variable Pair 1	{State Variable 1, State Variable 2}	Type II Test 1

4.8.2 Pre-Silicon Test Cases (Sigma Delta ADC)

One example validation test case for Sigma Delta ADC is shown in Table 17. The faulty circuit specification parameters are far off from nominal circuit ones. The faulty

circuit is created by varying design and process parameters of the circuit. 400 pairs of stimuli are designed for type I test and 100 stimuli are designed for type II test.

Table 17: Sigma delta ADC validation test case (process varied circuit)

	Nominal Circuit	Faulty Circuit
Specifications		
SFDR (dB)	30.44	22
THD (dBc)	-10.31	-2
ENOB (Bits)	4.76	3.36
Diagnosis		
Percentage of bits flipped triggering error		
Type I Test 1 (SV1)	0	81.0
Type II Test 1 (SV pair 1)	0	96.0

4.8.3 Pre-Silicon Test Cases (RF Receiver):

4.8.3.1 Test Case 1 &2: Faulty In Phase Mixer (Bias Voltage Variation):

In AMS/RF systems bias voltages/currents are generally controlled digitally. Two pathological test cases are created where bias voltage of the In Phase Mixer is varied (5% for test case 1 and 10% for test case 2) by supplying wrong digital codes. Other design and process parameters were not altered. Diagnostic results (see Table 18) shows that the state variables associated with In Phase Mixer causing more bit flips than the others.

Table 18: Pre-silicon validation results of RF receiver (*SV : State Variable)

	Percentage of bits flipped triggering error			
	Test Case 1	Test Case 2	Test Case 3	Test Case 4
Type I Test 1 (SV1)	5.1	5.3	11	12
Type I Test 2 (SV2)	52.1	68.0	83	25
Type I Test 3 (SV3)	23.1	30.1	80	34
Type I Test 4 (SV4)	4.8	5.0	13	15
Type II Test 1 (SV pair 1)	12.1	13.3	80	71
Type II Test 2 (SV pair 2)	14.8	13.8	83	68
Type II Test 3 (SV pair 3)	46.6	73.3	90	76

4.8.3.2 Test Case 3 &4: DC offset and Gain Variation:

We created two more pathological test cases 1) introduced DC offset in LNA output (test case 3) and 2) gain of In Phase Mixer is increased by changing design parameters (test case 4). Type I test fails in these two pathological cases. All type II tests catch these faults easily (see Table 18).

4.8.4 Post-Silicon Test Cases (RF Receiver):

Signal coupling, noise coupling and supply voltage variation form the majority of the electrical post-silicon bugs. Using two chains of RF receiver shown in Figure 60, a 2x2 MIMO receiver is formed. As shown in Figure 64, a coupling fault is forced by introducing a capacitance between LNA outputs of the chains. Conventional specification testing (EVM testing) will not catch this bug, although it will show up in actual operation and will corrupt received MIMO data. If EVM testing is done in SISO mode sequentially, the coupling bug will not be activated. Even if the two chains tested concurrently (required two sets of costly RF test instruments), the bug will not show up unless the two chains carry different symbols. How BISCC catches this bug is shown in Table 19.

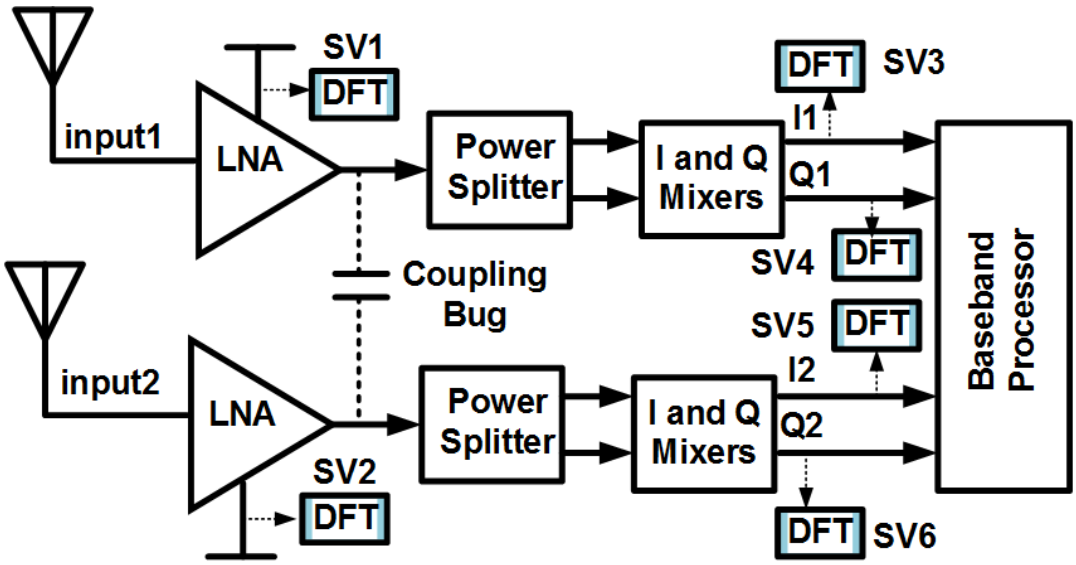


Figure 64: 2x2 MIMO receiver

Table 19: Post-silicon validation results of RF receiver (*SV : State Variable)

	Percentage of bits flipped triggering error
Type I Test 1 (SV2)	89
Type I Test 2 (SV4)	91
Type I Test 3 (SV6)	92

BISCC test methodology has been applied to various other kinds of circuits such as sigma delta ADC and DAC, analog biquad filter, analog comparator, sense amplifier etc. We have considered variety of fault cases, process parameter variation, temperature variation, resistive/capacitive opens and shorts, electrical bugs, logical bugs. In almost all cases BISCC has uncovered the fault with great accuracy.

4.8.5 Temperature Variation

The RF receiver system (Figure 60) is designed to operate in the temperature range 0^{O}C to 50^{O}C. We are showing two examples where the operating temperature is beyond the above said acceptable range. How the proposed built in assertion based diagnosis raises error flag in validation are shown in Table 20. It is clearly seen that the type I test 1 is not showing error while all the type II tests are indicating malfunctioning of the device at 100 and -20 ^{O}C temperature. In order to explain the above observation, we plot the observed state variable 1 for a random signal applied at the input of the RF receiver system in Figure 65. We see a clear gain compression/enhancement in the response of the captured state variable 1 for various temperatures (nature of the captured responses are similar only differing in amplitude values). It is explained previously why a type I test fails if the anomaly seen is due to only gain compression/enhancement.

Table 20: Post silicon validation results of RF receiver system at various temperatures

	Percentage of bits flipped triggering error		
System temperature	27 ^{O}C	100 ^{O}C	-20 ^{O}C
Type I Test 1 (State Variable 1)	0	1	4
Type I Test 2 (State Variable 2)	0	11	70
Type I Test 3 (State Variable 3)	0	12	67
Type I Test 4 (State Variable 4)	0	1	6
Type II Test 1 (State Variable pair 1)	0	63	75
Type II Test 2 (State Variable pair 2)	0	55	69
Type II Test 3 (State Variable pair 3)	0	66	73

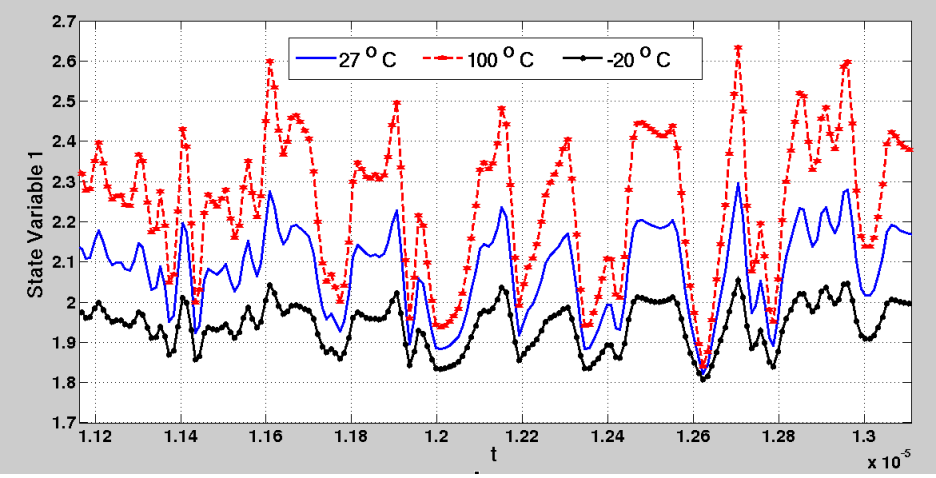


Figure 65: Captured state variable 1 for a random stimulus

4.8.6 Effects of Sampling Clock Jitter

As the proposed validation methodology is a self-checking scheme i.e. the circuit/system checks state consistency among its own states (temporally and spatially), one may be skeptical about the effects of sampling clock jitter in the performance of the proposed methodology. We run the following simulations (shown in Table 21 and Table 22) to verify the potency of the proposed methodology for random clock jitter. For type I test 1000 stimuli pairs are used, and for type II test 200 stimuli are used. Sampling clock frequency used is 10MHz (sampling clock period is 100ns). Average error trigger rate for type I test is 1.65% for 1ns random jitter and is 2.1% for 2ns random jitter.

Table 21: Effect of random clock jitter on nominal circuit's (RF receiver) state reachability for type I test

Random Clock Jitter (ns)	Error Trigger (%)		
	Type I		
	State Variable 1	State Variable 2	State Variable 3
0	0	0	0
1	1.6	1.7	1.6
2	2	2.1	2.2

Table 22: Effect of random clock jitter on nominal circuit's (RF receiver) state reachability for type II test

Random Clock Jitter (ns)	Error Trigger (%)		
	Type II		
	State Variable pair 1	State Variable pair 2	State Variable pair 3
0	0	0	0
1	0	0	2.5
2	1.9	2.6	2.0

4.8.7 Test Time Reduction

EVM testing of a transceiver takes about or more than 300ms [97]. We have used 1000 stimuli pair for type I test and 200 stimuli for type II test in manufacturing testing of the receiver. Each stimulus is of duration $0.5\mu s$, so the total test time required is 1.1ms ($2200 * 0.5\mu s$). Moreover EVM testing does not provide diagnosis capability. Though in this work we have not strived to do diagnosis and fault isolation, the methodology described is capable of doing that. We have kept diagnosis using state consistency as future work.

4.9 Conclusions and Future Work

The authors have presented BISCC, a novel low cost, quick validation technique for embedded AMS/RF systems. Observability is a key issue in post-silicon debug of deeply embedded analog/RF system. With on chip signature capturing and temporal and spatial signature comparing infrastructure, the authors present a built in self-validate methodology for RF/analog systems. BISCC is equally applicable to pre-silicon

hierarchical model equivalence checking between a high level design and a low level transistor netlist. Vdd ramping technique discussed in [91] for mixed-signal/RF validation is an orthogonal approach to the proposed scheme of this work. The authors would like to fuse Vdd ramping into the proposed scheme in future.

CHAPTER 5. CONCURRENT BUILT IN TEST AND TUNING OF BEAMFORMING MIMO SYSTEMS USING LEARNING ASSISTED PERFORMANCE OPTIMIZATION

5.1 Introduction

Research on 5G massive MIMO systems with large numbers of transmit and receive antennas is moving forward at an electrifying pace [98-102]. However, with increasing levels of circuit complexity and higher operating speeds, the underlying electronics becomes highly susceptible to manufacturing process variations. Going forward, next generation beamforming wireless systems will need to be designed with *built-in test and post-manufacture self-tuning capability for managing manufacturing yield and in-field tuning.* In addition, power consumption of high speed wireless communications systems is of increasing concern and must be factored into the tuning process.

Prior work on built-in self-test and self-tuning has focused mostly on "omnidirectional" SISO and MIMO wireless systems [103-106]. We propose to develop novel self-test and self-tuning algorithms for *beam-steering MIMO front end designs* that scale across beam-steering 5 – 73 GHz architectures (WiFi – mm-wave 10m indoor communication). The challenges relative to the state of the art are as follows:

1) Depending on the architecture of the beam steering MIMO system employed, signals are combined and up/down converted in the transmitter/receiver in different ways before processing in the baseband DSP through an array of data converters [98-102]. This introduces challenges regarding decoupling of test results for

individual antenna-RF chains from data obtained for combined signals (internal RF chain signals may not be externally observable).

2) A subsequent problem is to *concurrently test all mixed-signal/RF components in all the RF chains* with the least test cost (complexity of test signals needed, minimum test time). The test procedure must measure a diversity of RF performance specifications of each of the RF chains: *phase-shifter accuracy, I-Q mismatch, DC offsets and nonlinearities* of the various RF components as well as unwanted signal coupling across RF chains and local oscillator leakage. Moreover, the tests applied *must produce diagnostic information* to enable efficient post-manufacture tuning of the complete RF transceiver to offset the effects of silicon manufacturing process variations.

3) Beam-steering MIMO RF systems employ programmable phase-shifters that must be calibrated for all desired beam-steering angles. With active phase-shifters, a desired phase of the output signal of the phase-shifter relative to the phase of the input signal is generally difficult to achieve with constant gain across all phase values. This necessitates gain compensation using other components in the RF chain which in turn affects their phase transfer functions as well. Such phase-gain inter-relationships complicate tuning algorithms for the entire RF system. This is further made much worse by the fact that *these relationships are process-corner dependent* (i.e. vary from device to device) and any tuning algorithm must accommodate such dependencies. It is assumed that the gain/distortion/power consumption of individual RF components such as mixers, amplifiers and impedance matching networks are designed to be digitally tunable. Given the

above, tuning of all the RF chains of a massive-MIMO system needs to be performed with a *high degree of parallelism across all the RF chains* due to the large numbers of component-level tuning knobs involved. Such tuning is highly dependent on the speed and accuracy of the testing procedures discussed earlier and must ensure accuracy of specifications while minimizing power consumption of tuned devices. Further, it is necessary to guarantee that the maximum error vector magnitude (EVM) of received symbols [19, 20] as well as signal-to-interference ratio is within prescribed bounds across all the designated *beam-steering angles* of the MIMO transceiver.

The rest of the chapter is organized as follows: First, beamforming MIMO architectures are discussed in Section 5.2. This is followed by key contributions and approach in Section 5.3. In Section 5.4 and 5.5, the proposed parallel testing approach is discussed. In section 5.6 we discuss about models to translate RF impairments to EVM and SINR. In Section 0, concurrent (parallel) tuning algorithms are presented. This is followed by a discussion of the experimental results in Section 5.8. Finally, conclusions are presented in Section 5.9.

5.2 Beamforming MIMO Receiver Architectures

Beamforming receiver architectures are shown in Figure 66. In this work the individual analog/RF circuits (LNA,VGA, Phase Shifter, Mixer) are built using 45nm FreePDK models [45] . Digital baseband processing is simulated in computer.IQ vector sum type programmable phase shifter (as shown in Figure 67) is built. Phase shifting is governed by relative strength of I and Q current DACs (given in Eq. 56). The quadrant of

the phase shift is determined by the four switches SW1 to SW4. LNA and Mixer circuits are shown in Figure 68 and Figure 69 respectively. Variable gain amplifier circuit is shown in Figure 70. Detailed circuit design and control loop operations are described in [107].

$$\theta = \tan^{-1}\left(\frac{Q}{I}\right) \tag{56}$$

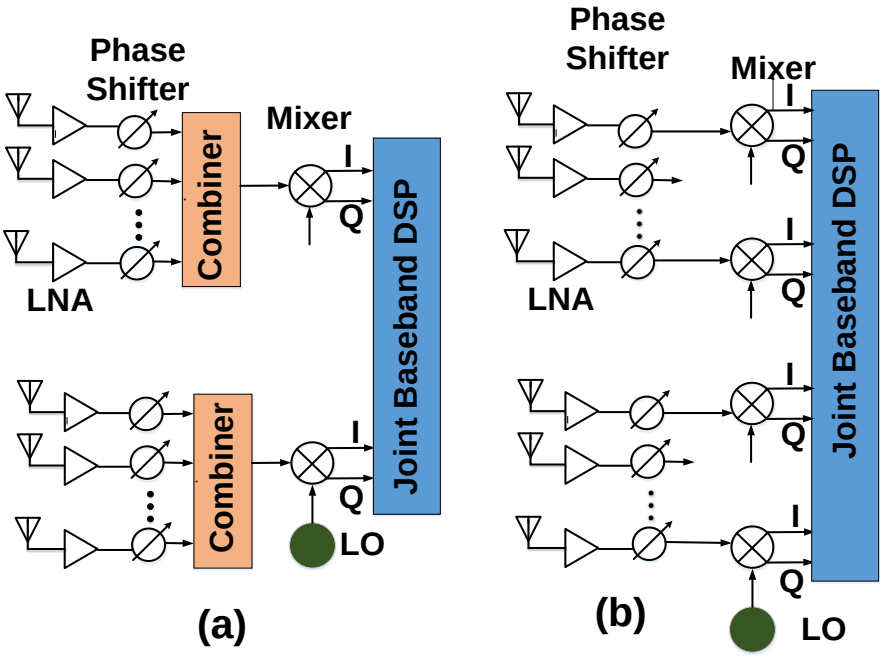


Figure 66: MIMO architectures (a) analog beamforming (b) digital beamforming

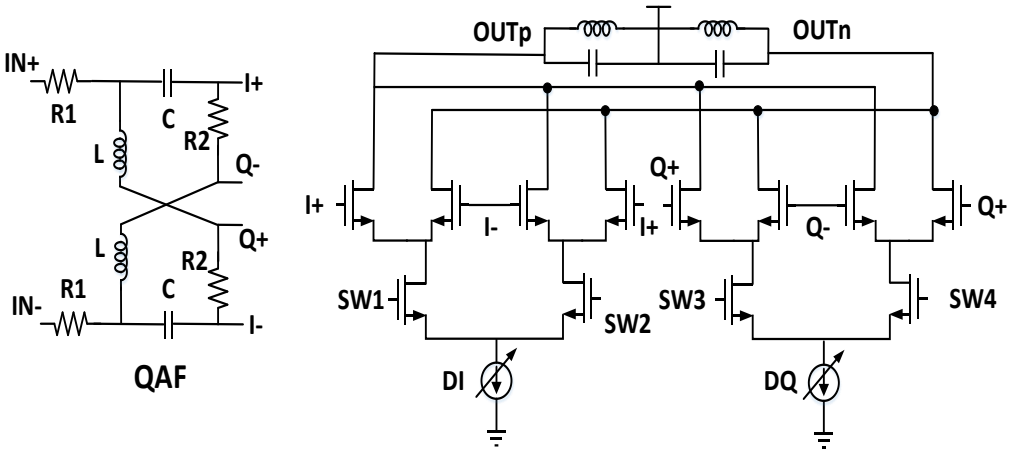


Figure 67: Programmable phase shifter

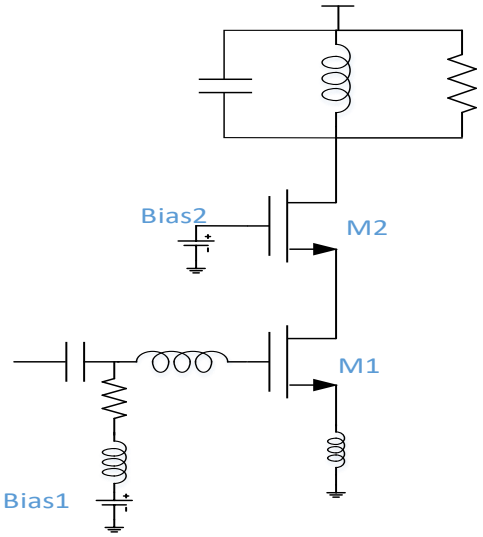


Figure 68: Cascode LNA

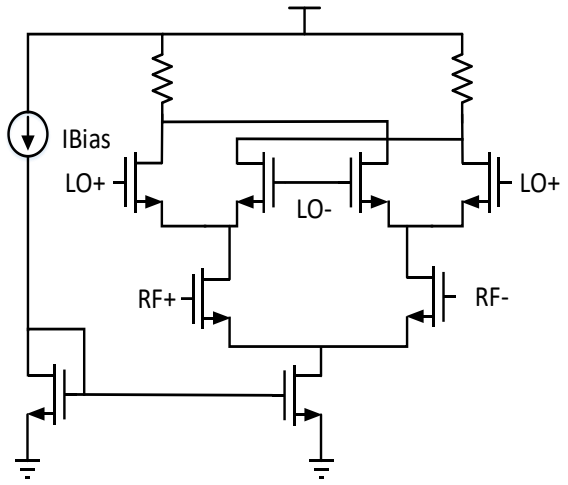


Figure 69: Gilbert Cell Mixer

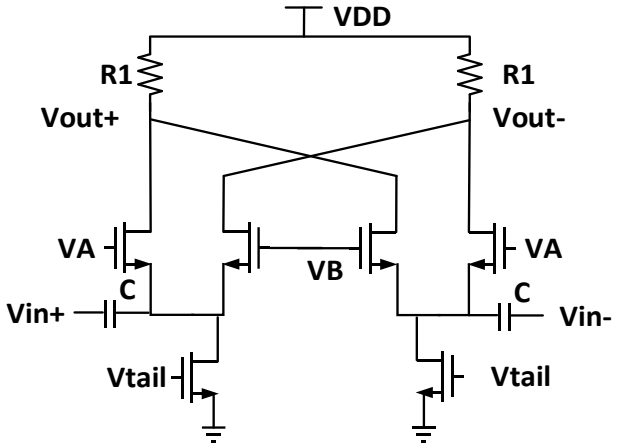


Figure 70: Variable gain amplifier

In a MIMO beamforming system, the signals arriving at various beamforming antennas traverse different distances as the antennas are physically separated from each other. The corresponding path differences manifest themselves as phase differences of the carrier for narrowband received signals. Figure 71 shows one example where beamforming antennas are arranged in a straight line and the consecutive antennas are separated by a distance of d. If the line of sight incidence angle is θ then the phase difference of the

received carrier between two consecutive antenna elements is given by Eq. 57 (ω is the carrier frequency and c is the speed of light).

$$\phi = \frac{\omega d}{c} cos\theta \tag{57}$$

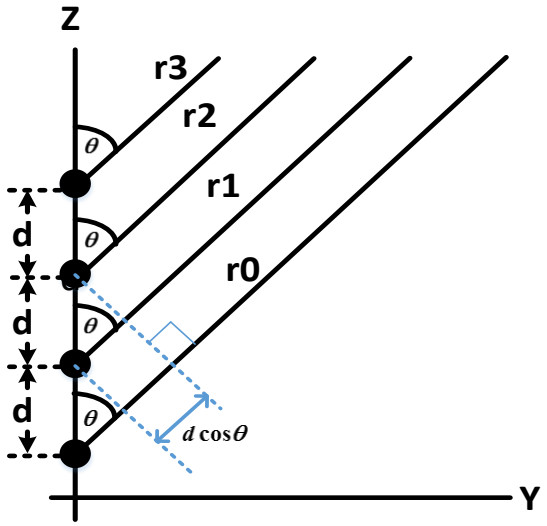


Figure 71: Beamforming

For a given incident angle (θ) beam steering attempts to constructively add the received signals and destructively add the received signals for all other incident angles. One example beamforming receiving is shown in Figure 72. There are a range of possible MIMO beamforming architectures [99, 102]. These span, analog, digital (Figure 66) and hybrid beamforming architectures. Receiver systems can be all-RF, heterodyne with phase shifting in the main RF path and heterodyne with LO phase shift [99]. The use of digital beamforming imposes very tight linearity constraints on the design of the RF components. For these reasons, analog beamforming architectures are attractive. However, the use of a combiner (summation unit) as shown in Figure 66 (a) for an analog beamforming receiver,

significantly complicates the testing of individual RF chains whose outputs are no longer directly observable as in digital beamforming systems (Figure 66 (b)).

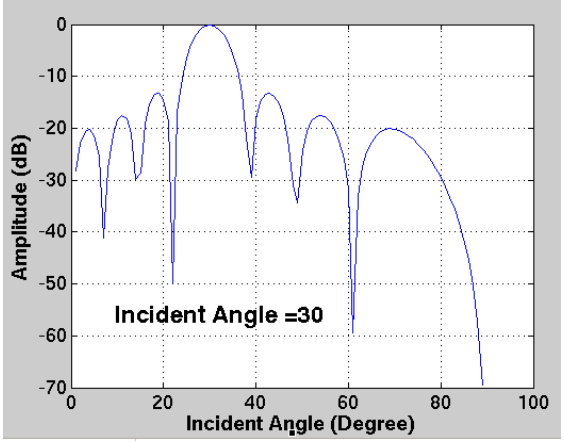


Figure 72: Antenna array factor for beamforming

5.3 Approach and Key Contributions

The proposed approach comprises of collaborative testing and tuning algorithms targeted towards analog beamforming architectures. These are harder to test than digital beamforming systems from a test observability perspective. Without loss of generality, the solutions produced are easily ported over to corresponding digital beamforming architectures. The steps involved are described as follows:

Step 1. High resolution parallel phase and gain testing:

First, parallel phase error testing algorithms that determine the phase and gain errors of different RF chains corresponding to *N* different beam steering angles with high accuracy, are developed. For practical beam steering systems, it is seen that *N* tests applied in parallel to all the RF chains are sufficient to characterize phase and gain mismatches in

the RF chains. A *key contribution is that the proposed solution addresses the loss of RF chain output observability due to the use of signal combiners* as shown in Figure 1 (a). Consequently, this allows beamforming transmitters to be also tested using identical testing algorithms by observing only the combined PA outputs of the transmitter chains involved. We focus on non-idealities of the RF chain itself and assume that the fidelity of the antenna is guaranteed by design (it is difficult to measure electromagnetic antenna radiation patterns during in-socket manufacturing test).

Step 2. Parallel testing of RF chain non-idealities:

Next, two techniques are presented:

(a) Testing for distortion effects: In this approach, the *combined effects* of parallel phase-shifted RF chains of a beam-steering system on the received or transmitted signal are computed. This requires coprime input frequencies so that intermodulation of the various chains do not overlap. This technique is suitable for small number of beamforming chains. A better frequency efficient technique is explained below.

(b) Frequency-efficient parallel testing of individual RF chains: In this approach, the *distortion introduced into the received or transmitted signal by individual RF chains* can be directly measured using multiple test tones in a frequency-efficient manner while observing only the down-mixed output of the combiner of Figure 1 (a). While prior techniques [108, 109] apply test tones to parallel RF chains in ways that enforce frequency separation in the responses produced by the different chains, we relax this requirement, allowing tones resulting from distortion in different chains to overlap with each other. This allows larger numbers of chains to be tested in parallel (higher frequency efficiency) for a

specified signal bandwidth, while allowing the distortion specifications of each chain to be determined individually as in prior testing schemes.

Step 3. Mapping of test results to EVM (error vector magnitude) and SINR (signal-to-interference ratio):

The test results obtained in steps 1 and 2, above are mapped to EVM/SINR performance metrics using a mapping mechanism driven by simulation with an end-calibration performed using experiments on hardware to account for test instrumentation imperfections. This is a modification of the techniques for EVM measurement presented in [110, 111].

Step 4. Test result driven parallel tuning of beamforming MIMO systems:

Parallel tuning is performed in two phases:

(a) *Phase 1: Learning assisted coarse tuning*: Each tuning knob of the RF system consists of 8 bits of tuning control (8 bit current DAC). The higher order 4 bits are determined in Phase 1. This is achieved using a version of the "one-shot" tuning algorithms presented in [104, 108, 109]. Here, the test results from Step 2 are used to predict the tuning knob values corresponding to each RF chain using supervised learning driven algorithms that map the test responses directly to the optimal tuning knob control bit values. Since the tests are run in parallel, such tuning can also be performed in parallel. A key difference is that the prior algorithms need to be modified to handle multiple beam steering angles of the design.

(b) Phase 2: Gradient descent driven fine tuning: Fine tuning of the lower order 4 tuning control bits is performed using iterative testing and tuning algorithms driven by

gradient descent based on a cost function optimization metric that includes total power consumption. This is also performed in parallel for all the RF chains concerned supported by the parallel nature of the testing algorithms.

Note that the procedures discussed above, can be applied to testing and tuning of analog and digital beamforming transmitters as well, where the outputs of all the transmitter PAs are combined and observed during manufacturing test. Also, the proposed methods can be applied to hybrid beamforming architectures with some modifications. For ease of explanation and for the sake of brevity, we focus our discussion on the testing of analog beamforming receivers and use the same as a test vehicle for demonstrating our ideas and approach. In the following we discuss each of the steps of the proposed approach above.

5.4 High Resolution Parallel Gain/Phase Testing

For determining the relative phases of N beamforming RF chains, N tones with frequencies that are coprime to each other are selected. For example, for the system of Figure 73, with $N=4$, sinusoidal signals of frequency f1,f2,f3 and f4 are chosen so that they are co-prime to each other. Each of the signals is modulated with the carrier frequency generated by a local oscillator (LO: on-board for built-in test, external otherwise) and applied to the respective receiver LNA inputs as shown in Figure 73. The Fast Fourier Transform (FFT) of the received signal is computed by the baseband processor. As the input signals are co-prime to each other, harmonic distortion in one chain does not affect the accuracy of measurements for other RF chains as long as the n'th order harmonics fall in different FFT frequency bins of the baseband response signal. The spectrum of the

received signal corresponding to a test designed for the receiver of Figure 73, (4 RF chains) is shown in Figure 74.

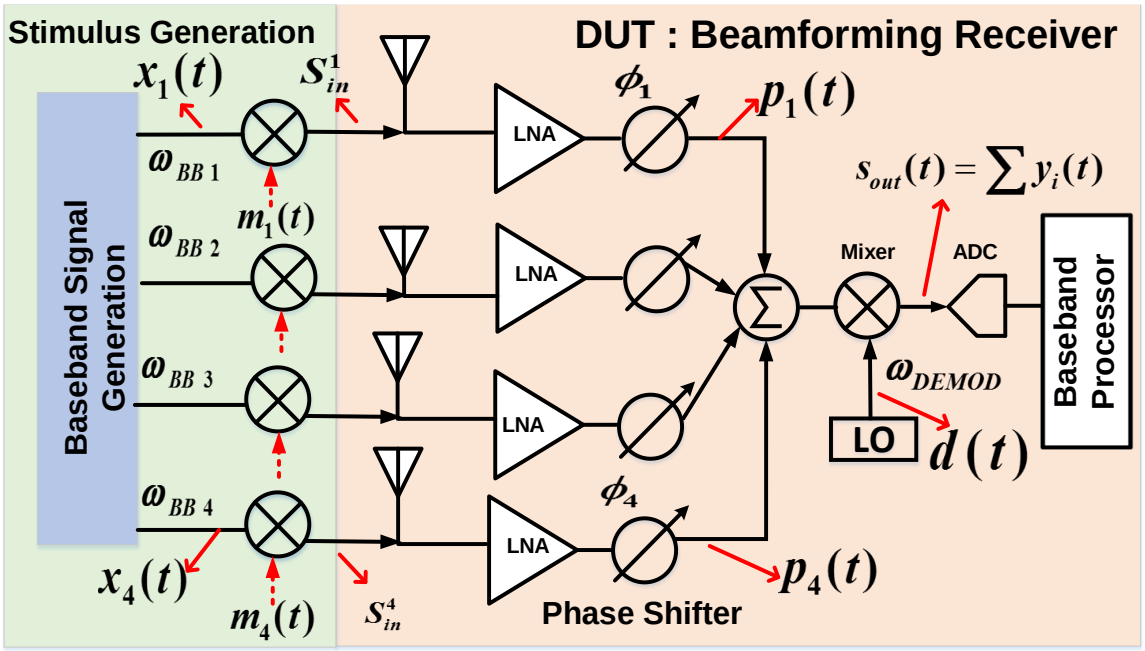


Figure 73: MIMO receiver characterization (analog beamforming)

In general, where OFDM systems are involved, each tone in an OFDM frame can be used as a test tone. For example, for an OFDM frame with 64 tones, the number of channels that can be tested concurrently is the number of co-prime integers between 5 and 64 and is equal to 17.

The input baseband, modulating and demodulating signals of Figure 73, are given in Equations 58,59 and 60 respectively.

$$BaseBand\ Signal: x_i(t) = A_{BBi}Sin(\omega_{BBi}t) \quad (58)$$
$$Modulating\ Signal: m_i(t) = A_{MODi}Sin(\omega_{MODi}t) \quad (59)$$
$$Demodulating\ Signal: d(t) = A_{DEMOD}Sin(\omega_{DEMOD}t) \quad (60)$$

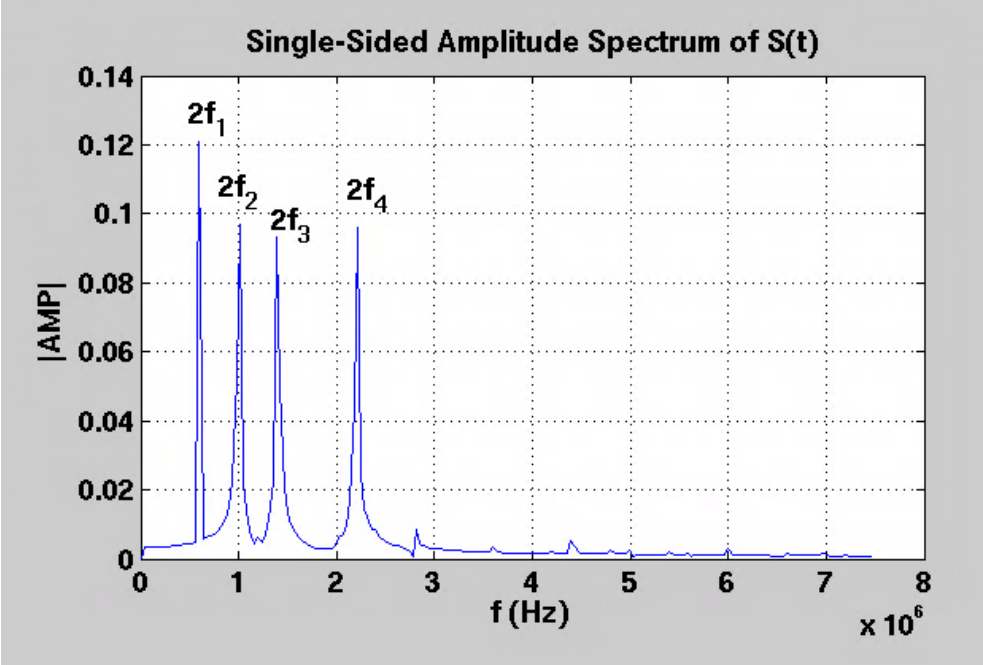


Figure 74: FFT of received signal

The signals $p_i(t)$ that are input to the summation unit of Figure 73 are given by Eq. 61.

$$p_i(t) = A_{MODi}A_{BBi}Sin(\omega_{MODi}t + \phi_i)Sin(\omega_{BBi}t) \tag{61}$$

Consequently, the signal $S_{out}(t)$ can be derived as $\sum_{i=1}^{4} A_{DEMOD}Sin(\omega_{DEMOD}t) * p_i(t)$. If we denote $y_i(t) = Sin(\omega_{DEMOD}t) * p_i(t)$, then $y_i(t)$ can be written as shown in Eq. 62.

$$y_i(t) = M_i Sin(\omega_{MODi}t + \phi_i)Sin(\omega_{BBi}t)Sin(\omega_{DEMOD}t) \tag{62}$$
$$where\ M_i = G_i A_{MODi}A_{BBi}A_{DEMOD}\ G_i: path\ gain\ of\ i^{th}\ chain$$

This further simplifies to $y_i(t)$ as shown in Eq. 63 which can be further reduced to the form shown in Eq. 64.

$$y_i(t) = 0.5M_i\{cos(\omega_{MODi}t - \omega_{BBi}t + \phi_i) - cos(\omega_{MODi}t + \omega_{BBi}t + \phi_i)\}\sin(\omega_{\mathrm{DEMOD}}t) \quad (63)$$

$$\begin{aligned} y_i(t) = 0.25M_i\{&Sin(\omega_{DEMOD}t + \omega_{MODi}t - \omega_{BBi}t + \phi_i) \\ &+Sin(\omega_{DEMOD}t - \omega_{MODi}t + \omega_{BBi}t - \phi_i) \\ &-Sin(\omega_{DEMOD}t + \omega_{MODi}t + \omega_{BBi}t + \phi_i) \\ &-Sin(\omega_{DEMOD}t - \omega_{MODi}t - \omega_{BBi}t - \phi_i)\} \end{aligned} \quad (64)$$

Removing high frequency components from Eq. 64 we get Eq. 65.

$$\begin{aligned} y_i(t) = 0.25M_i\{&Sin((\omega_{DEMOD} - \omega_{MODi} + \omega_{BBi})t - \phi_i) \\ &-Sin((\omega_{DEMOD} - \omega_{MODi} - \omega_{BBi})t - \phi_i)\} \end{aligned} \quad (65)$$

If we choose $(\omega_{DEMOD} - \omega_{MODi}) = \omega_{BBi}$ then Eq. 65 reduces to Eq. 66. The frequencies ω_{MODi} and ω_{BBi} within the external test instrumentation are selected appropriately to meet this condition.

$$y_i(t) = 0.25M_i\{Sin(2\omega_{BBi}t - \phi_i) - Sin(-\phi_i)\} \quad (66)$$

Removing DC parts from Eq. 66 we get Eq. 67 (low pass filters present in the circuit remove DC components).

$$y_i(t) = 0.25M_iSin(2\omega_{BBi}t - \phi_i) \quad (67)$$

Eq. 67 above, depicts the received signal for a single antenna chain. The combined signal corresponding to all four antenna chains of Figure 73, is given by Eq. 68.

$$s_{out}(t) = \sum_{i=1}^{4} y_i(t) = 0.25\sum_{i=1}^{4} M_iSin(2\omega_{BBi}t - \phi_i) \quad (68)$$

For gain measurement, even in the presence of high non-linearity and asymmetric phase shifts in the different RF chains, the received signal amplitudes are very accurate. However, phase measurement suffers from FFT quantization error. The FFT provides a

very good guess of the respective phase differences between the various chains of the receiver but accurate phase estimation requires further steps as discussed below.

By selecting the test input frequencies judiciously, high frequency amplitude distortion and phase shift of the carrier is transferred to low frequency components. Consequently, the zero crossing points of the tones concerned can be measured with timing sensitivity enhanced by the ratio of the LO (carrier) frequency to the frequency of the baseband tone. This allows sub-degree RF chain phase shift measurements with high accuracy. From Eq. 68, it is apparent that the received baseband signal frequencies are known. Amplitudes (A_i) and phases (ϕ_i) of the input baseband signals are distorted by the RF circuits. $S_{reconstructed}$ (shown in Eq. 69) is the reconstructed signal in the baseband where frequencies (ω_{BBi}) are known, while the amplitudes (A_i) and phases (ϕ_i) are unknown (variables). The problem is to generate accurate estimates of the unknown variables above.

$$\begin{aligned} S_{RECONSTRUCTED}(t) = \{&A_1 Sin(2\omega_{BB1}t - \phi_1) \\ &+ A_2 Sin(2\omega_{BB2}t - \phi_2) \\ &+ A_3 Sin(2\omega_{BB3}t - \phi_3) \\ &+ A_4 Sin(2\omega_{BB4}t - \phi_4)\} \end{aligned} \tag{69}$$

A rough estimate of A_i and ϕ_i is obtained by taking the FFT of the received signal. A simple signal reconstruction based optimization as shown in Eq. 69 is used to tune the values A_i and ϕ_i, until the time-domain waveform ($S_{reconstructed}$) corresponding to specified values of the same matches the digitized waveform at the output of the ADC of Figure 73 ($S_{received}$). This further improves the resolution of amplitude and phase measurement. The optimization problem is stated as:

$$minimize \ | S_{received}(t) - S_{reconstructed}(t) | \text{ s.t.} \quad (70)$$
$$L_m A_i^{FFT} < A_i < U_m A_i^{FFT}$$
$$\phi_i^{FFT} - \phi_m < \phi_i < \phi_i^{FFT} + \phi_m \quad for \ i = 1,2,3,4$$

where A_i^{FFT} is the amplitude of the i^{th} channel baseband signal from the FFT, ϕ_i^{FFT} is the phase of the i^{th} channel baseband signal from the FFT, $L_m A_i^{FFT}$ and $U_m A_i^{FFT}$ are lower and upper search limits of the i^{th} channel signal amplitude and ϕ_m is the allowed search margin from the measured phase. In this work, we assume $\phi_m = 5^O$ $L_m = 0.9$ and $U_m = 1.1$.

The problem above, described in Eq. 70, is solved by an interior point trust region gradient descent optimizer [112] (fmincon function of Matlab is used). As the search space is narrowed by apriori FFT measurements, the optimization converges to minima rapidly. We have performed this optimization with various initial seed solutions and found that the optimizer always finds the same solution. This points to the objective function being locally convex within the algorithm search space.

We conducted an experiment in which we arbitrarily set the phase tuning knobs of the receiver chains. First we measured the gain and phase difference of each chain independently. This serves as the golden reference for comparison against the respective values determined by our algorithm. Then we measured gain and phase difference

concurrently for all chains using the procedure described above. The accuracy of the proposed concurrent phase and gain measurement algorithm is shown in Table 23.

Table 23: Measurement accuracy
(A: $Amplitude\ in\ Volt$ ϕ: $phase\ in\ degree$)

	Sequential measurement (Golden)		Parallel measurement	
			FFT	FFT and Optimization
Channel 1	A	0.126	0.122	0.124
	ϕ	5.260	5.200	5.190
Channel 2	A	0.126	0.106	0.128
	ϕ	23.30	26.01	23.35
Channel 3	A	0.126	0.106	0.128
	ϕ	48.37	68.20	48.26
Channel 4	A	0.126	0.106	0.128
	ϕ	69.54	67.78	69.53

5.5 Parallel Testing of RF Chain Non-idealities

5.5.1 Testing for Distortion Effects

In this section, we discuss concurrent measurement of non-linearity related parameter (IIP3, P1dB, and IIP2) across all RF chains. If we apply a two-tone ($f_{i1}\ and\ f_{i2}$) modulated stimulus $S_{in}^{i}(t)$ to the i^{th} RF chain with amplitude A_{in}^{i} (as shown in Eq. 71), the fundamental and intermodulation tones at the output of the mixer (after demodulation) are as shown in Figure 75. The 3rd order intermodulation terms ($2f_1 - f_2\ \ and\ 2f_2 - f_1$) result from gain compression and the 2nd order intermodulation terms are caused by mixer spurious response [44].

$$S_{in}^{i}(t) = \{A_{in}^{i} sin(2\pi f_{i1} t) + A_{in}^{i} sin(2\pi f_{i2} t)\} \sin(2\pi f_{MODi} t) \quad (71)$$

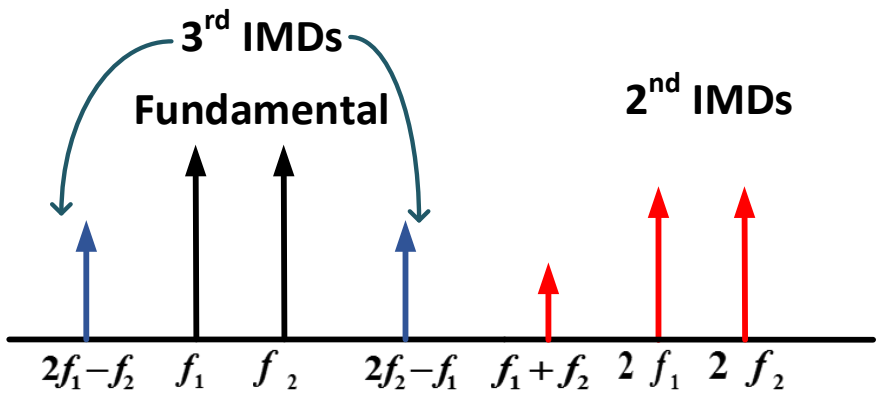


Figure 75: Applied two tone and intermodulation tones

The FFT of the demodulated output signal provides the amplitude and phase of each frequency component (shown in Figure 75). From FFT measurement, the IIP3 of the channel and IP2 of the mixer can be obtained from Eq. 72 and Eq. 73 respectively.

$$IIP3_{channel} = A_{in} \sqrt{\frac{V_{f1}}{V_{2f1-f2}}} \quad (72)$$

$$IIP2_{mixer} = A_{in} \sqrt{\frac{V_{f1}}{V_{f1+f2}}} \quad (73)$$

Let us assume α_1^i and α_3^i are the gain and 3rd order intermodulation coefficients of channel i. The IIP3 and P1dB values in terms of α_1 and α_3 are given in Eq. 74 and Eq. 75 respectively. From gain and IIP3 measurement, α_1 and α_3 of the channel can be extracted and hence P1dB can be derived in accordance with Eq. 75.

$$IIP3_{channel} = \sqrt{\frac{4\alpha_1^i}{3\alpha_3^i}} \tag{74}$$

$$P1dB_{channel} = \sqrt{0.145\frac{\alpha_1^i}{\alpha_3^i}} \tag{75}$$

If the intermodulation and fundamental tones of the two-tone stimulus applied to each RF chain do not overlap, then the FFT of the demodulated signal can be used to determine all the channel non-linearity parameters concurrently. However, this requires test signals across a wide frequency band proportional to the number of chains being tested concurrently and limits test efficiency for massive-MIMO systems with large numbers of parallel RF chains. To resolve this, we propose a "frequency-efficient" parallel testing approach which does not require such frequency separation (i.e. some intermodulation tones corresponding to different RF chains can overlap). This is described next.

5.5.2 Frequency Efficient Parallel Testing

Consider the i^{th} RF chain of Figure 73, for which the large signal representation of $y_i(t)$ is given by Eq. 76. We assume that the phase ϕ_i of the chain is determined using small signal analysis of $y_i(t)$ as described earlier in Eq. 62- 68. The frequencies and phases of all the tones generated in the output $y_i(t)$ (given by Eq. 76) corresponding to the i^{th} RF chain, due to nonlinearities in the chain after filtering out all the high frequency components are given in Table 25 (we assume up to 3rd order nonlinearity in the RF chain). In Eq. 76, $(\alpha_{1i}, \alpha_{2i}, \alpha_{3i}, \beta_{1i}, \beta_{2i})$ define the gain and intermodulation coefficients of the i^{th} RF chain.

$$y_i(t) = (\alpha_{1i} S_{in}^i(t) + +\alpha_{3i} {S_{in}^i}^3(t))(\beta_{1i} S_{LO} + \beta_{2i} S_{LO}^2) \quad (76)$$
$$S_{LO} = sin(2\pi f_{DEMOD} t)$$

Assume k RF chains are testing in parallel with the i^{th} chain stimulated with two tones at two orthogonal frequencies: $(2i-1)f$ and $2if$, $1 \le i \le k$, respectively. Table 24 gives all the frequencies generated due to nonlinearities in the i^{th} RF chain due to the stimulus described above (Eq. 71). In Table 24, f is the lowest frequency tone over all the RF chains. The problem is to calculate the amplitude of each tone generated by the nonlinearities in each RF chain (given in columns 3-7 of Table 24) given i) the combined amplitude and phase of the tones produced by each RF chain ($y_i(t)$ in Eq. 76), ii) the amplitudes of test tones applied and the phases of (see Table 25) corresponding intermodulation terms to each RF chain.

We first show that the individual amplitudes A1 and A2 of two tones can be computed if their combined amplitude A, phase ϕ and individual phases ϕ_1 and ϕ_2 are known. This is shown in Eq. 77 and Eq. 78 resulting in two equations that can be solved uniquely for the two variables A1 and A2.

$$A sin(2\pi f t + \phi) = A_1 \sin(2\pi f t + \phi_1) + A_2 \sin(2\pi f t + \phi_2) \quad (77)$$
$$= \sqrt{(A_1^2 + A_2^2 + 2A_1 A_2 \cos(\phi_2 - \phi_1))}\, sin\left(2\pi f t + \arctan\left(\frac{A_1 sin\phi_1 + A_2 cos\phi_2}{A_1 cos\phi_1 + A_2 sin\phi_2}\right)\right)$$

$$A = \sqrt{(A_1^2 + A_2^2 + 2A_1 A_2 \cos(\phi_2 - \phi_1))} \quad (78)$$
$$tan(\phi) = \frac{A_1 sin\phi_1 + A_2 cos\phi_2}{A_1 cos\phi_1 + A_2 sin\phi_2}$$

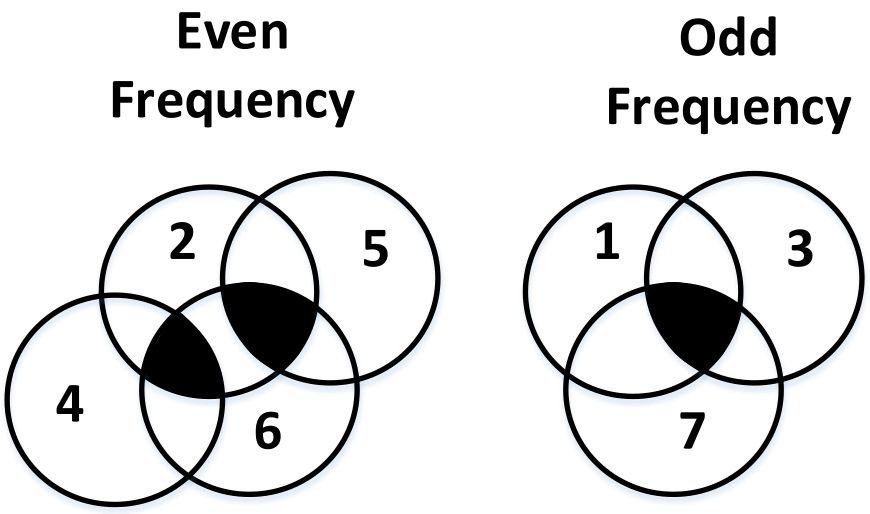


Figure 76: Frequency overlap Venn diagram

Table 24: Frequency components at the receiver output in presence of 2nd and 3rd order distortions

channel	Col 1	Col 2	Col 3	Col 4	Col 5	Col 6	Col 7
	f_1	f_2	$f_1 + f_2$	$2f_1$	$2f_2$	$2f_1 - f_2$	$2f_2 - f_1$
1	1	2	3	2	4	0	3
2	3	4	7	6	8	2	5
3	5	6	11	10	12	4	7
4	7	8	15	14	16	6	9
5	9	10	19	18	20	8	11
6	11	12	23	22	24	10	13
…	..	..	..	..	..	..	..
k	2k-1	2k	4k-1	4k-2	4k	2k-2	2k+1

It is seen (see Figure 76) that for every frequency generated in Table 24, there are contributions from at most 3 RF chains, one of which is a fundamental tone applied to an RF chain (whose amplitude is known). The frequencies in columns 1,3 and 7 of Table 24, are always odd. So, any odd frequency overlap from three channels will always have

column 1 frequency (fundamental) component. Frequencies in column 2,4,5,6 are always even. Frequencies in column 4 and 5 are mutually exclusive, they cannot overlap. For any integer value of k1 and k2, 4k1-2 and 4k2 will never overlap. So, for even frequencies also, frequency overlap from three channels will always have column 2 frequency (fundamental) component. For example, the frequency 4f consist of an input tone (fundamental) to RF chain 2, is produced by the 2f2 component of RF chain 1 and produced by the 2f2-f1 component of RF chain 3. Hence, the amplitude and phase of the measured tone at frequency 4f will consist of the summation of all three tones as described above. If a single frequency in Table 24 corresponds to two generated tones, the individual amplitudes of the two tones can be computed using Eq. 77 and Eq. 78. If a single frequency of Table 24 corresponds to three tones and if one of the tones is an input frequency (fundamental) to an RF chain, then the amplitudes of the remaining two non-fundamental tones can be calculated by applying Eq. 77 and Eq. 78 in two steps: (a) Calculate A_m and ϕ_m from$Asin(2\pi ft + \phi) = A_m\, sin(2\pi ft + \phi_m) + A_{in}^{i} sin(2\pi ft + \phi_0)$, where $A_{in}^{\mathrm{i}}\sin(2\pi ft + \phi_0)$ is the input tone (fundamental) to the i'th RF chain concerned and (b) Calculate A1 and A2 from $A_m sin(2\pi ft + \phi_m) = A_1 \sin(2\pi ft + \phi_1) + A_2 \sin(2\pi ft + \phi_2)$. Once the amplitudes of all the tones corresponding to the k RF chains of Table3 are computed, the IIP2, IIP3 and P1db specifications of all k RF chains are calculated.

Table 25: Phase of each frequency component (ϕi : channel phase shift)

frequency	phase
f_1	ϕi
f_2	ϕi
$2f_1 - f_2$	$\phi i + \pi$
$2f_2 - f_1$	$\phi i + \pi$
$f_1 + f_2$	$2\phi i + \frac{\pi}{2}$
$2f_1$	$2\phi i + \frac{\pi}{2}$
$2f_2$	$2\phi i + \frac{\pi}{2}$

Accuracy of FFT based measurement depends on sampling rate of the acquired signal. For a given ADC sampling rate (see Figure 73), there is a frequency difference limit on adjacent two tones to be accurately detected by FFT. Hence number of frequency bins in a given baseband bandwidth is limited. How many frequency bins are available for testing in a given band can be found by drawing analogy with available sub-carriers in an OFDM band. A 20MHz OFDM band has total 64 sub-carriers, among which 52 are available for data transmission [113]. So $\delta f = 0.3125MHz$ is sufficient to distinguish two adjacent tones. Using the above information, Bandwidth savings for the proposed scheme over the non-overlapping scheme is shown in Figure 77. For 16 channels 20MHz is sufficient for the proposed scheme, whereas a 57.5MHz bandwidth is required for non-overlapping scheme.

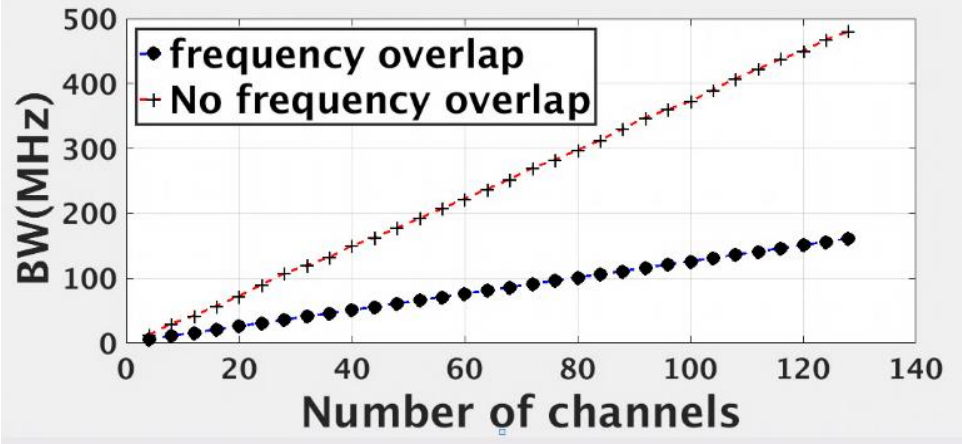


Figure 77: Bandwidth requirements comparison

5.6 Mapping of Test Results to EVM

In the previous section, we have shown how to measure all the RF impairments of all the chains concurrently form two tests. In this section, we will briefly discuss the model to translate RF impairments to EVM and SINR. EVM, SINR and power are the system level metrics used to quantify the performance of a RF system. Tuning SISO RF systems based on EVM is shown in [114]. As these tuning processes are run for multiple iterations and evaluating EVM is time consuming, post manufacture tuning based on EVM is not suitable from cost perspective. In [111] the authors developed an analytical model of EVM based on AM to AM, AM to PM,IQ mismatch and IIP3 of an OFDM transmitter. In [110]

the authors have developed a regression based EVM prediction model based on static and dynamic RF impairments. In this work, we have adopted the idea of EVM and SINR prediction from RF impairments and tune for RF impairments which is essentially tuning for EVM and SINR. RF impairments we have used to model our RF system are gain, phase, non-linearity, IQ mismatch and LO feed through. As in beamforming MIMO, we have 100's of tuning bits, relationship between each tuning bit and EVM is complicated and there is aliasing among tuning bits which makes tuning optimization intractable. On the contrary relationship of tuning bits to RF impairments are modular, for example tuning bits to tune gain and IIP3 of the LNA is independent of phase in phase shifter. For reliable MIMO transmission space time block coding [115] is used. EVM of a MIMO system generally points to combined EVM of all the MIMO channels. In manufacturing tuning individual EVM requirements of a MIMO channel can be evaluated from combined EVM requirement [109]. Here EVM model shown below is for single beam forming MIMO channel (multiple beamforming chains).

5.6.1 EVM model

A symbol S and its modulated version S_{mod} are shown in Eq. 79 and 80 respectively.

$$S = I + jQ \tag{79}$$

$$S_{mod} = Isin(2\pi ft) + Qcos(2\pi ft) = Asin(2\pi ft + \psi) \tag{80}$$

$$where\ A = \sqrt{(I^2 + Q^2)}\ \ and\ tan\psi = Q/I$$

After phase compensation (phase shifting in the receiver chain) S_{mod} becomes S_p (shown in Eq. 81)

$$S_p^i = Asin(2\pi ft + \psi + \phi_i) \tag{81}$$

Non-linearity of the LNA transform S_p to S_n (Eq. 82). Signals are summed at the adder (Eq. 83).

$$S_n^i = \alpha_1^i S_p + \alpha_3^i S_p^3 \quad (82)$$

$$S_{sum} = \sum S_n^i \quad (83)$$

After demodulation and low pass filtering the summed signal, received symbol is obtained (Eq. 84 and 85).

$$I_{received} = S_{sum}\,(\beta_1^I LO_I + \beta_2^I LO_I^2) \quad (84)$$

$$Q_{received} = S_{sum}\left(\beta_1^Q LO_Q + \beta_2^Q LO_Q^2\right)$$

$$where\ LO_I = \sin(2\pi ft)\ and\ LO_Q = \cos(2\pi ft)$$

$$S_{received} = I_{received} + jQ_{received} \quad (85)$$

EVM definition is given in Eq. 86.

$$EVM = 100 * \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(S^i - S_{received}^i\right)^2}}{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(S^i)^2}} \quad (86)$$

5.6.2 SINR model

Array factor of a beamforming system is defined in Eq. 87. In this work $\phi_{interference} = \phi_{in} + 30^O$ is considered. SINR is defined in Eq. 88.

$$AF = \sum AF_i = \sum_{i=1}^{n} f(A)e^{-j(\phi_{in}^i - \phi_{PS}^i)} \quad (87)$$

$$\phi_{in}: Input\ signal\ phase\ \phi_{PS}: phase\ shift\ in\ the\ chain$$

$$f: nonlinearity\ transformation\ function\ explained\ in\ previous\ section$$

$$SINR = \frac{AF_{Signal}}{\sum AF_{interference}} \quad (88)$$

5.7 Test Data Driven Parallel Tuning

As the tuning is learning assisted, first we need to build a model from known devices to assist tuning in the later stage. In order to create an efficient learning model, we need to sample devices from diverse process corners. The sampling technique to be used for device selection is described below.

5.7.1 Device selection criteria for software model

We assume that all the process parameters are normally distributed and the joint probability density function (pdf) of the process parameters is given by Eq. 89, below.

$$f(x_1, x_2 \ldots\ldots, x_n) = f(x) \tag{89}$$
$$= \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp(-(x-\mu)^T \Sigma^{-1} (x-\mu))$$
$$where\ x_1, x_2, \ldots . x_n\ are\ n\ process\ parameters$$
$$\mu = [\mu_1\ \mu_2 \ldots \mu_n]\ mean\ of\ process\ parameters$$
$$\Sigma\ is\ covariance\ matrix\ of\ process\ parameters$$

To create device instances corresponding to diverse process corners, a simple strategy is to generate devices by sampling the joint probability density function described by Eq. 89. However, this creates a large number of device instances centered around the mean of the joint pdf above. For model learning to be effective, we need instead many devices around the test specification acceptance boundaries of the DUT. To force this bias, we generate device instances using a uniform distribution of the process parameters as shown in Figure 78, bounded by its 3σ limits. For every sampled device from the uniform sample space a weight is associated which is a measure of the probability of the device instance being generated if the process space was sampled from the joint normal distribution (Eq. 89). For a single process variable, this measure is the area under the

normal curve around the sampled point (an example is shown in Figure 78 for x=-1). If the sampled device corresponds to the process parameters $(x_1', x_2', \ldots., x_n')$ the corresponding weight w for the corresponding device instance is given by Eq. 90.

$$w = \iiint_{\substack{x_1=x_1'-k*\sigma_1 \\ x_n=x_n'-k*\sigma_n}}^{\substack{x_1=x_1'+k*\sigma_1 \\ x_n=x_n'+k*\sigma_n}} f(x_1', x_2', \ldots., x_n')dx_1dx_2 \ldots.dx_n \quad (90)$$

where k is a constant $k = 0.1$ *is chosen in this work*

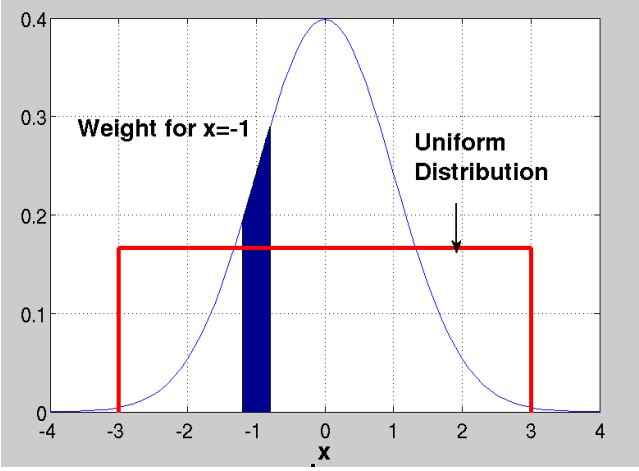


Figure 78: Probability weight for single variable

To ensure that device instances are selected from diverse process corners, a large number N of device instances are sampled as per the uniform process parameter distribution of Figure 3. Only a limited number M of N device instances are used. To select such M of N devices, first K random transient stimuli are generated. Subsequently, each of the N devices is stimulated by the K random test patterns. Consequently, every device instance is associated with K response vectors corresponding to time-sampled values of the response waveform. The distance d_{ij} between any two response vectors R_i and R_j corresponding to different devices i and j is given by the L2 norm of R_i and R_j(Eq. 91).

The distance between two device instances *i* and *j*, D(i,j) is defined to be the largest distance across all the K random test stimuli (Eq. 92). To identify the M devices out of N, K-means clustering is performed to cluster the N devices into M clusters in such a way that the mean distance between all devices in a cluster is minimized.

$$d_{ij} = \frac{1}{N}\sum_{t=1}^{N}\left(R_{it} - R_{jt}\right)^2 \tag{91}$$

$$D(i,j) = \max\ {d_{ij}}^k \tag{92}$$

$$where\ d_{ij}^k\ is\ the\ distance\ between\ response\ vectors\ R_i and\ R_j for\ k^{th}\ stimulus$$

The last step of the procedure consists of picking one device instance from each cluster to generate the M devices. Let L be the vector of test specification limits for each of the test specifications of the DUT and let L_u be the corresponding vector of test specification values for the *u'th* device instance. We determine the device u in each cluster with the smallest norm $||L-L_u||$ (this is the device that is closest to the test acceptance limits of the DUT). When two or more devices have similar values of $||L-L_u||$, the device with the higher weight w, defined by Eq. 90 is selected from the cluster of device instances.

Table 26: Device selection algorithm

1.	*N: Number of randomly generated devices;*
2.	*M: Number of devices to be selected.*
3.	*Generate K random transient test stimuli;*
4.	*Simulate all N device instances and capture response signatures corresponding to all K stimuli for each device;*
5.	*For all pairs of devices i, j, compute the distance D(i,j);*
6.	*Use k-means clustering algorithm to partition the devices into M clusters:*
7.	*Pick one device from each cluster based on its distance from the test specification limits, resolving conflicts via its weight "w" as per equation* 90.

5.7.2 Device Selection Criteria for Manufactured Hardware Devices:

For manufactured ICs, the process parameter values are not known. Only transient response is available from the devices. Similar clustering algorithm (shown in Table 26) is adopted here, conflicts are resolved by randomly choosing any one of the candidate devices.

5.7.3 Two Stage Tuning Methodology

In this section, we will describe a learning based process corner identification technique to be used in mimo system tuning. As shown in Figure 80, using the algorithm described in previous section devices are sampled from population of devices. These sampled devices are fully characterized (for all tuning knob settings) using the characterization technique described earlier. These devices are tuned using the fully characterized surface plot of performance and tuning knobs. In production testing phase, it is impractical to tune all the devices using their fully characterized surface plot, as generating these surface plots (performance versus knob settings) takes long time, especially for architectures with large number of tuning bits. In this work, we envisioned to split the tuning into two phases (see Figure 79), crude tuning (quickly setting the higher order control bits) and fine tuning (setting the lower order tuning bits). While fine tuning takes care of inter die variation, coarse tuning accounts for intra die process variation.

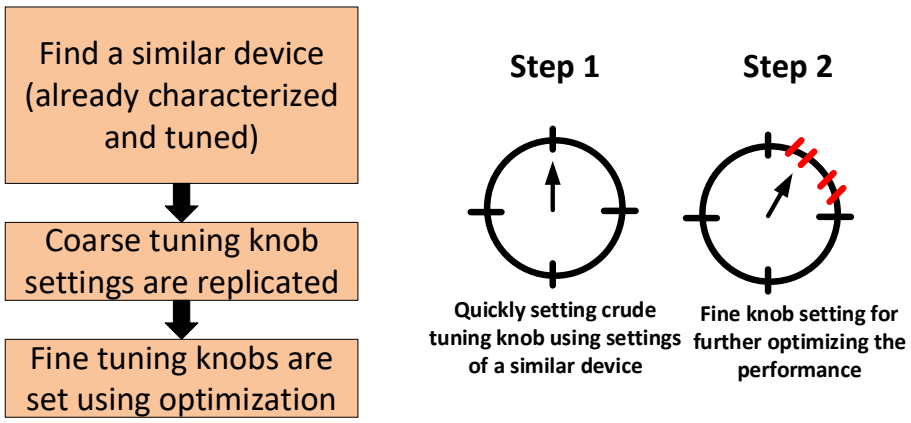


Figure 79: Two step tuning

5.7.3.1 Coarse Tuning:

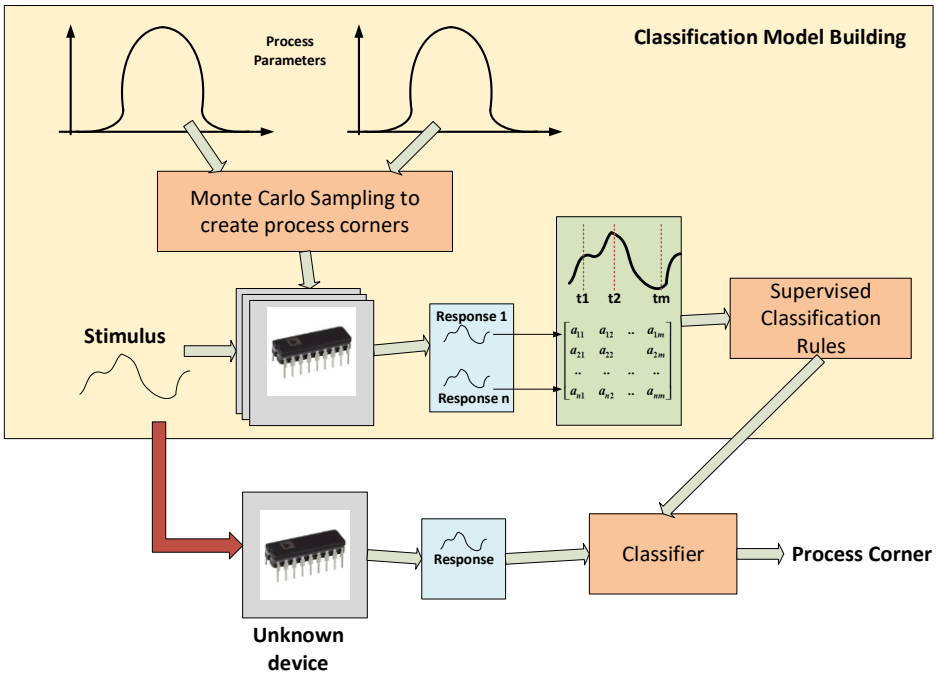


Figure 80: Process corner identification and first cut tuning (setting coarse bits from a similar device)

Coarse tuning procedure is pictorially shown in Figure 80 and briefly described in Table 27.

Table 27: First cut coarse tuning steps

Step 1: select representative devices from population	Representative devices are selected from pool of available devices. (a)In simulation stage Monte Carlo sampling on process parameters will create different process corner devices. (b)In post-production stage representative devices are so selected that each produces a unique signature. (Details of this algorithm is given in Table 26)
Step 2: Characterizing and tuning representative devices.	These representative devices will be characterized and tuned manually for minimizing power and maximizing EVM and SINR.
Step 3: Stimulus generation	Based on these sampled devices a stimulus generation program a framed so that these device responses are maximally separated.
Step 4: Classification rule forming	Based on the above stimulus and device responses, a PNN is trained for classification purpose.
Step 5: tuning new device	The stimulus from step 3 is applied, response is captured and using the PNN formed in step 4, the new device is classified into one of the process corner devices. All the coarse bits of the phase gain and non-linearity controllers of the RF system will be replicated from the already tuned similar device.

In Figure 80 we have shown that a stimulus is used to generate classification rules based on sampled devices and the same stimulus is used to excite a DUT to classify it based on the previously formed classification rules. Now we will describe the algorithm to synthesize that stimulus.

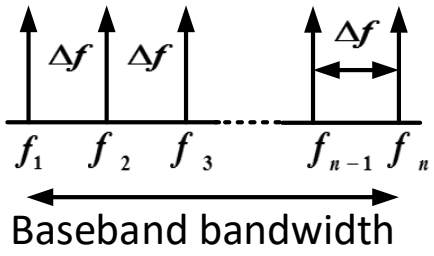


Figure 81: Baseband spectrum

We choose n number of frequency bins from the baseband bandwidth such that adjacent bins are equidistant from each other (as shown in Figure 81). An optimum stimulus is a combination of judiciously selected frequencies from the above said bins, such that the response corresponding to the stimulus is maximally sensitive to process parameter perturbation. Genetic algorithm was used to synthesize such a stimulus from a pool of candidate stimulus (as shown in Eq. 93).

$$\text{obj: min } miss\ classification\ rate \tag{93}$$
$$candidate\ stimulus: \frac{1}{k}\sum_{i=1}^{n} I_i A_i \sin(2\pi f_i t)$$
$$where\ k: normalization\ factor, I \in \{0,1\}\ , A \in [0,1]$$

5.7.3.2 Fine Tuning Procedure

After finding a similar device and replicating the coarse tuning bits from it, surface plots (fine tuning knob vs RF impairments) around that chosen coarse tuning position are used in fine tuning procedure (as shown in Figure 82). These surface plots are not accurate as they are coming from a similar device not from the exact device under test. Had the surface plots been available for the DUT, it would have been one shot tuning (all the tuning knobs are set in one iteration). These surface plots provide the gradient in the gradient based search process and thereby guide the search process. At every iteration in the search process the RF impairments are measured using the techniques described earlier and using the models EVM and SINR are predicted. Iteration steps are shown in Table 28. Phase mismatch has the largest effect on EVM and SINR so we tune phase first. Then we tune

for gain and non-linearity and again tune for phase as step 2 and 3 changes phase to some extent.

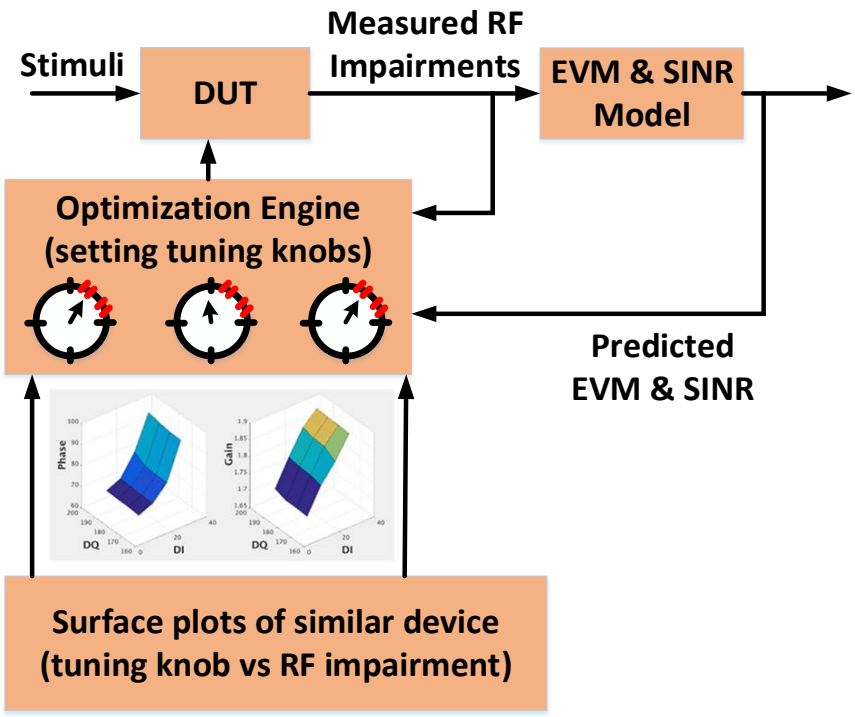


Figure 82: Fine tuning procedure

Table 28: Fine tuning steps

Step 1: phase tuning	Tune phase shifters to minimize phase offset $\min \sum_{i=1}^{N} \lvert \phi_{iincident}^{i} - \phi_{channel}^{i} \rvert$
Step 2: tune VGA's	Tune VGA's so that gain of every chain is identical $\min \sum_{i=1}^{N-1} \lvert g_N - g_i \rvert$ $where\ g_N : gain\ of\ Nth\ chain$
Step 3: tune LNA & Mixer	Tune LNA and Mixer to improve linearity
Step 4:	Repeat step 1 , tune for phase

5.8 Experimental Results

Following the sampling criteria mentioned in previous section process parameters (vth,tox,W,L) were varied (+- 15%) to create process varied devices. We created 2000 devices, 1000 devices are used in model building and rest 1000 devices are used as unknown DUTs to be tuned. Optimized stimulus for coarse tuning and similar device prediction is shown in Figure 83.

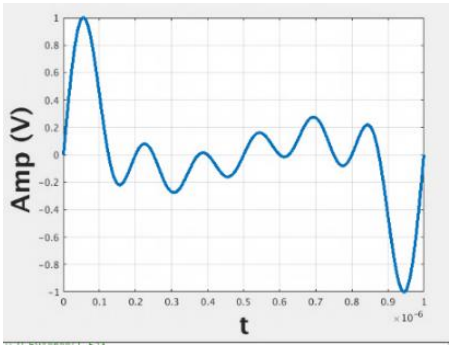


Figure 83: Optimized stimulus

Surface plot of phase shifter and VGA (for an individual receiver chain) w.r.t controlling digital bits are shown in Figure 84 and Figure 85 respectively.

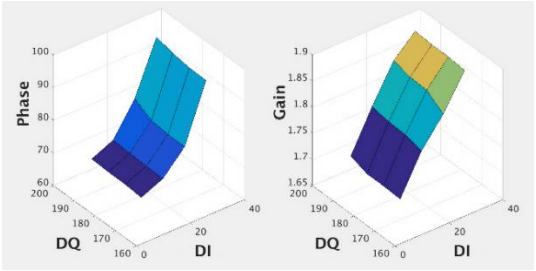


Figure 84: Phase and gain plots for a characterized phase shifter

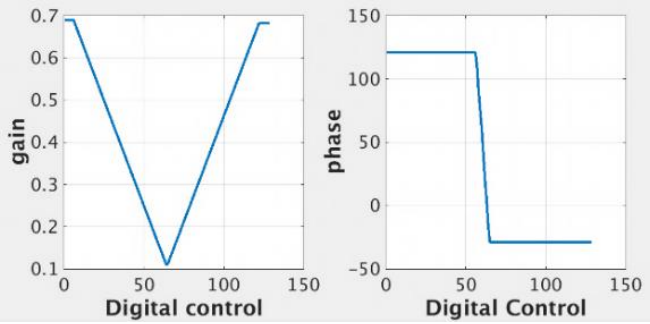


Figure 85: Phase and gain plots for a characterized VGA

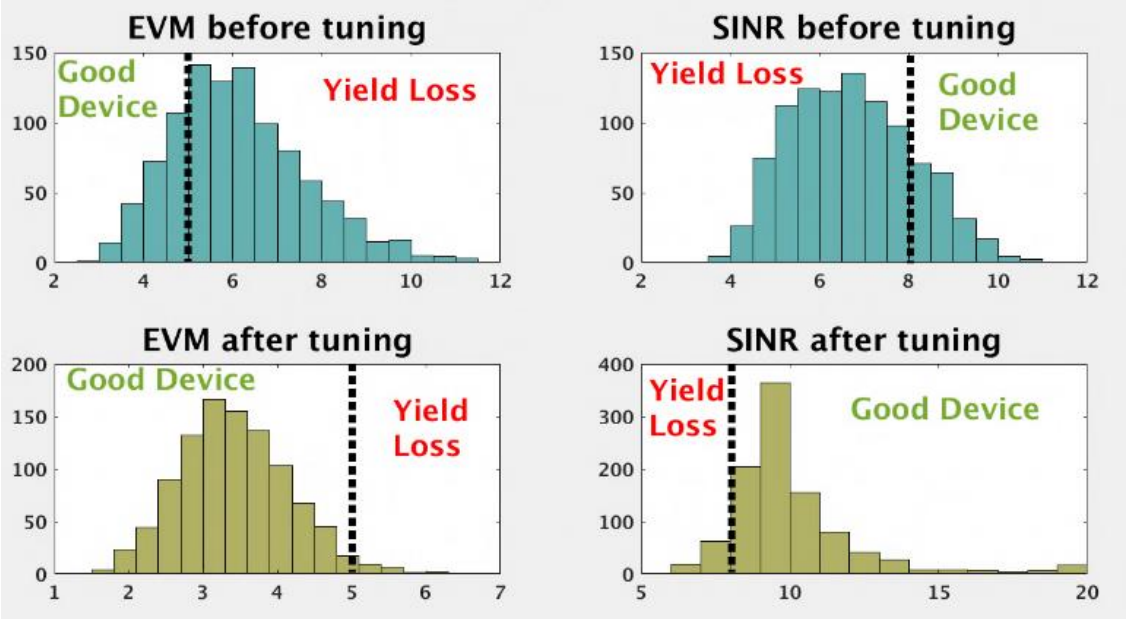


Figure 86 : Distribution of devices before and after tuning

Efficacy of the above-mentioned tuning procedure is shown by an experiment where we take 1000 beamforming mimo receivers from diverse process corners and tune them. Acceptance boundary for EVM and SINR are set at 5% and 8dB respectively. Distribution of devices and corresponding yield loss are shown in Figure 86. Yield is improved from 11% to 89% after tuning. Total number of tuning bits per beamforming chain is 32 (VGA: 8 LNA: 8 Phase Shifter: 16). 4 chain beamforming receiver will have 128 tuning bits, and 2x2 MIMO beamforming receiver will have 256 tuning bits. The

authors are not aware of any other state of the art tuning methodology where such a huge number of tuning bits are set concurrently. The whole optimization process takes 2-3ms in MATLAB.

5.9 Conclusions and Future Work

In this work the authors have shown a novel on chip parallel frequency efficient testing procedure for MIMO beamforming systems. All the chains of the beamforming systems can be tested and tuned concurrently, leads to significant savings in manufacturing testing time of MIMO 5g systems. In this work the authors have demonstrated the testing and tuning methodology for MIMO receiver, in future the authors would like to extend the procedure to MIMO transmitters.

CHAPTER 6. DESIGN OF ANALOG PHYSICALLY UNCLONABLE FUNCTION FOR SECURE COMPUTATION AND IC AUTHENTICATION

6.1 Introduction

Secure computing in insecure environment has emerged as one of the major research topics in the recent past. With the advent of cloud computing, Internet of Things (IOTs), and proliferation of smart computing devices (smart phones, tablets, smart TVs, game-consoles, e-readers etc.), the security of smart devices has become a major concern as a majority of these smart devices are operated in insecure environment. Until recently, security concerns were mainly handled in software. Hardware enforced security offer better protection than software only solutions [116]. Physically Unclonable Functions (PUFs) have emerged as one of the major hardware security primitives in recent times.

6.1.1 Current and Future Applications of PUF in Security:

The use of smart cards at present is ubiquitous. From banking and telecommunication applications, it has now forayed into electronic passports, electronic IDs, anti-counterfeiting devices, smart grid applications and many more [117]. Storing an authentication key inside smart card IC, makes smart cards and NFC enabled communication (electronic wallet) vulnerable to security threats. Generating keys on the fly by a PUF is heavily used in today's smart card and radio frequency identification (RFID) tag applications [118]. In the future PUF will likely also be used to protect external memory [117]. With the advancement of the Internet *of Things* (IOTs) and *cloud*

computing, the need for hardware device authentication and data encrypting/decrypting is extremely large. PUFs are an excellent fit for generating and hiding the authentication signature or cryptographic key for IOT and *cloud computing*. Low cost implementation of PUFs will make it a strong contender for next generation bar code. PUFs can also be used in software licensing, replacing hardware dongles used now a days [119].

Physical one way functions (POWF) [120] and physical random function [121] were precursors to Physically Unclonable Functions. Operation of PUFs is predicated on any physical parameter that varies randomly in IC manufacturing. The reported physical parameters that have been exploited to build PUFs are as follows: 1) delay of logic paths (arbiter, ring oscillator PUF) [122], 2) SRAM start-up behavior (SRAM PUF) [97], 3) glitches in digital circuitry (Glitch PUF) [123], 4) Sub-threshold transistor current fluctuation due to threshold voltage variation [124], 5) matrix material doped with random dielectric particles (coating PUF) [125], 6) cross coupled circuit elements (Butterfly PUF) [126], 7) power distribution system equivalent resistance variation [127]. Due to random dopant fluctuation (RDF), threshold voltage of a transistor shows spatially uncorrelated variability [128, 129]. In the sub-threshold region of operation current and threshold voltage of a transistor are exponentially related (random variability is exponentially multiplied). In this work we will exploit the above mentioned variability in a differential amplifier operating in subthreshold region to build the PUF.

The key benefits of the proposed PUF design are as follows:

1) Uniqueness of the designed PUF is better than that of an arbiter based PUF by a factor of 2X.

2) Reliability of the designed PUF is comparable to that of an arbiter based PUF.
3) Reverse engineering of the designed PUF is extremely difficult. (model building machine learning attack)
4) *An infinite number of challenge response pairs (CRPs) can be formed for the proposed PUF, making it a strong PUF.*

The remaining part of the chapter is arranged as follows: The proposed PUF architectures and operation is explained in section 6.2. Source of randomness (spatial and temporal) is explained in section 6.3. A brief overview of key generation and IC authentication using PUF is given in section 6.4. Challenge engineering concepts for generating suitable challenge response pairs are discussed in section 0. Simulation results, corroborating the idea proposed in this work are shown in section 6.6. Analog PUF performance metric for comparison among various analog structures is proposed in section 6.7. How the proposed analog PUFs (described in section 6.2) can be modified and used as public PUF is explained in section 6.8. Finally conclusions are drawn in section 6.9.

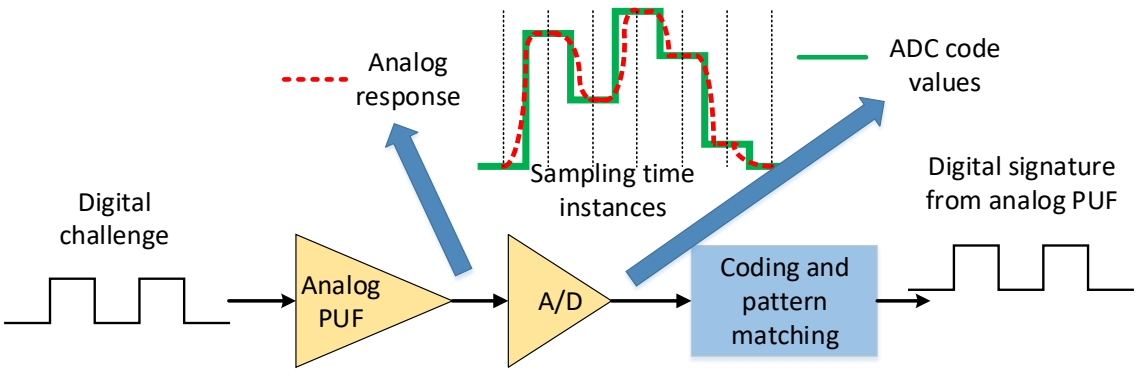


Figure 87: Signature generation by an analog PUF

6.2 PUF Architecture and Operation

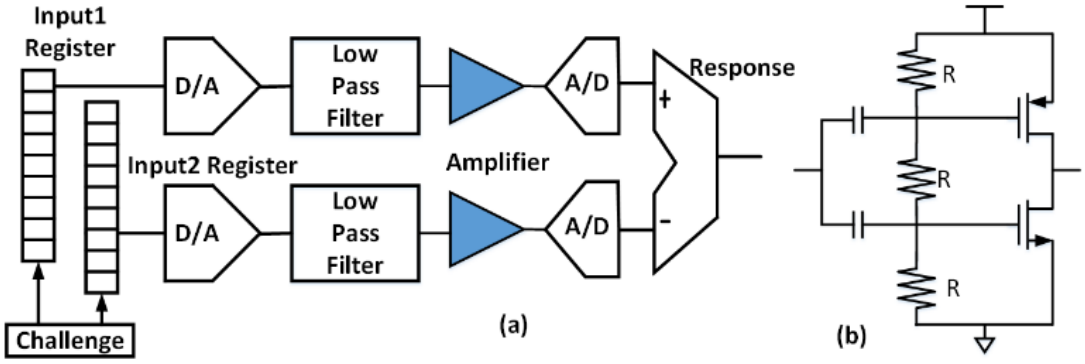


Figure 88: (a) PUF architecture 1 (b) Push pull amplifier

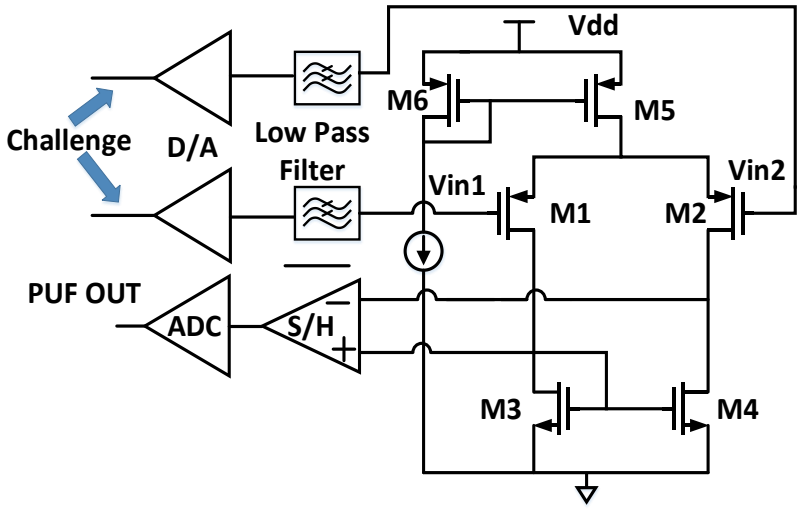


Figure 89: PUF architecture 2 (sub-threshold differential amplifier)

The proposed PUF architectures are shown in Figure 88 and Figure 89 . In architecture 1, the basic functional block is a push pull amplifier while in architecture 2, the basic functional block is a differential amplifier operated in sub-threshold region. Though the PUF operation is implemented by an analog circuit, its input and output are digital, so that it can be used in conventional PUF applications for key generation and IC authentication, without much change in peripheral circuitry (see Figure 87). Weak PUFs

are used in key generation where only one challenge response pair (CRP) is sufficient and adversary has no access to impart and see various CRPs. For IC authentication strong PUFs are used, whose CRPs are infinite in number. The most widely used PUFs for key generation are SRAM PUFs and the most widely used PUFs for IC authentication are arbiter and its variants (XORed arbiter, feed forward arbiter). SRAM PUF is a weak PUF as it can only generate one response (initial power up cell values). In a weak PUF, after provisioning (discussed in detail in section 6.4), helper data generation is blocked by burning fuses, so an attacker cannot apply challenges to it and observe responses. For strong PUFs on the other hand, as an attacker can apply multiple challenges and observe the responses, it can be subjected to model building machine learning attacks. In an arbiter PUF as the delay of each stage is additive with respect to the final output, a linear separator (such as support vector machine) can predict model of it using only a few 1000's of challenge response pairs. Non linearity in response makes model building difficult. In an arbiter PUF, the non-linearity can be introduced by XORing the output of several PUFs. Model building attacks on XORed PUFs are also reported in [130]. Device nonlinearity can also be used to amplify differences between two devices close to each other in the process/device parameter space thereby significantly increasing the uniqueness of the signature obtained from the PUF [124].

6.2.1 Architecture 1:

The PUF architecture proposed in this work is shown in Figure 88(a). The challenge bits are stored in the input registers. In every clock cycle, an 8 bit digital to analog converter produces an analog voltage from the stored challenge bits. These analog voltages are passed through a low pass filter and applied to an amplifier which is part of the PUF design.

The amplifier output is digitized by a 3 bit digitizer. Each PUF contains two amplifier chains as described above and the difference of the codes produced by the two digitizers concerned is considered as the PUF response. It should be noted that the input bit streams for the two chains are not identical. The amplifiers of Figure 88(b) are push pull amplifiers with no compensation feedback. Generally, analog amplifier transistors are large in size (high W/L ratios) for better noise performance. Here our objective is to have large variation in amplifier transfer characteristics with minimal change in the manufacturing process. Hence, transistor sizes are kept to a minimum. For the same reason, no feedback compensation circuitry is used to stabilize each amplifier. The key focus here is not to design an excellent amplifier, but to design an excellent PUF.

6.2.2 Architecture 2:

With regard to the proposed PUF of Figure 89, the current voltage relationship for a transistor in subthreshold region is exponential (as shown in Eq. 94). A small change in threshold voltage will create an exponential change in drain current. This exponential relationship between drain current and threshold voltage in subthreshold region is leveraged in this work as source of non-linearity.

$$I_d = I_s 10^{\frac{v_{gs}-v_{th}}{s}} \left(1 - 10^{-\frac{nv_{ds}}{s}}\right) \tag{94}$$

Where $I_s = 2n\mu C_{ox}\frac{W}{L}\left(\frac{KT}{q}\right)^2$ is nominal current, $s = \frac{nKT}{q} ln10$ is subthreshold slope, v_{ds} is drain to source voltage of a transistor. Differential voltage output expression is given by Eq. 95.

$$v_{out} = g_{m1}R_{out1}v_{in1} - g_{m2}R_{out2}v_{in2} \tag{95}$$

Where $R_{out1} = g_{d1}||g_{d3}$ and $R_{out2} = g_{d2}||g_{d4}$ and transconductance parameters g_m and g_d are defined as follows in Eq. 96 and 97 respectively.

$$g_m = \frac{\delta I_d}{\delta V_{gs}} = \frac{I_d \ln(10)}{s} \tag{96}$$

$$g_d = \frac{\delta I_d}{\delta V_{ds}} = \frac{I_d \ln(10)}{(10^{\frac{nv_{ds}}{s}} - 1)s} \tag{97}$$

From the above equations it is apparent that for a small mismatch in threshold voltages of differential pair transistors (M1 and M2 in Figure 89) there will be an appreciably large current imbalance in branches of differential amplifier. This current imbalance will cause large change in differential voltage as g_m and g_d are both strong functions of drain current and threshold voltage. Threshold voltage and dimensional (width and length of transistors) change of other transistors (M3, M4, and M5) will also contribute to change in differential voltage, although not as heavily as in differential pairs. Variation in M5 will change tail current and variation in M3 and M4 will affect branch currents. How the threshold voltage mismatch between differential pair transistors (M1 and M2) for different applied input voltages (related to PUF challenges) affect circuit response, is shown in surface plot of Figure 90 . 10% mismatch between M1 and M2 can cause full swing (rail to rail) change in circuit responses. As the response of the circuit (Figure 89) for applied challenges is highly non-linear, it can thwart model building machine learning attacks on this PUF.

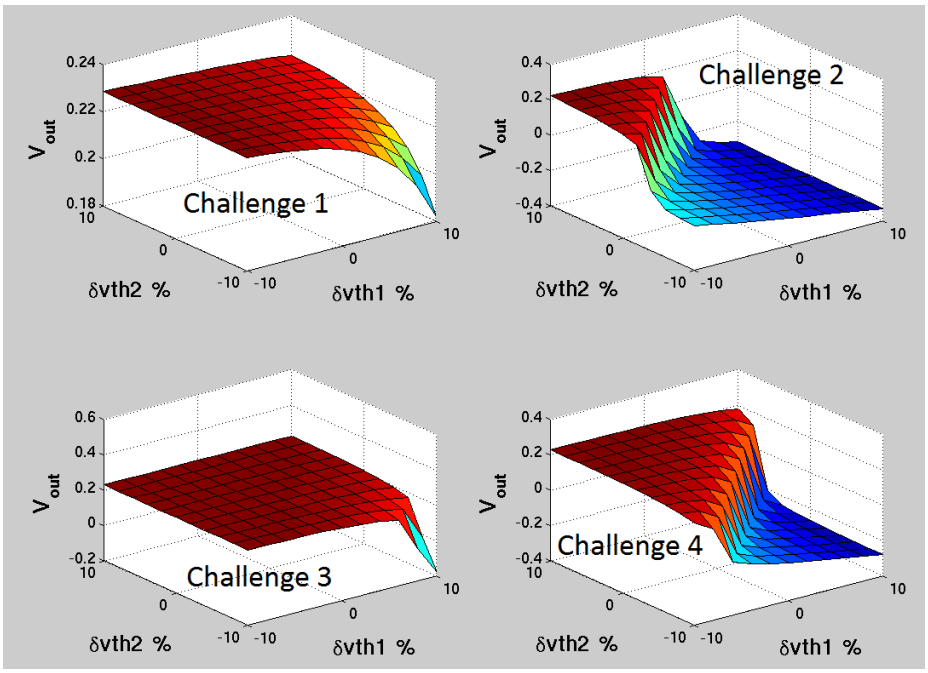


Figure 90: Variation of output voltage v/s percentage change in threshold voltages of differential pair transistors for four different challenges

6.3 Source of Randomness:

There has been research on harnessing randomness from manufacturing process variation by building various types of circuits that amplify the prevalent random process variation to circuit output voltages. No two transistors built on the same chip behaves identically. Threshold voltages of transistors in particular shows spatially uncorrelated variability due to random dopant fluctuations (RDF) [128, 129] .RDF is more pronounced in smaller channel devices. We also keep the transistor sizes minimum, to leverage variability from line edge roughness. Due to the presence of parasitic capacitances, analog circuits suffer from memory effects (hysteresis). Previous PUF designs were mostly digital, harnessing randomness from spatial process variation. This work is intended to fuse the best of both worlds. The analog structure proposed in Figure 89 is modified and a new

structure that fuses the both spatial randomness and analog hysteresis is shown in Figure 91(for architecture 2). A similar structure exploiting spatial and temporal randomness for architecture 1 is shown in Figure 93. The challenge bits are split into two groups' digital bits and analog bits. Digital bits are used to select any one differential pair, out of available differential pairs (spatial randomness). Analog bits are low pass filtered and converted to analog signal by a DAC and the resulting signal is applied to differential pair transistors of the sub-threshold amplifier. The differential voltage is sampled by a sample and hold circuit and digitized by a 3bit ADC. The amount of hysteresis present in any amplifier is dependent on the data rate (the frequency of random bit stream) and output capacitance. The proposed PUF is operated at 20MHz data rate and the hysteresis behavior is observed at various capacitive load conditions. Figure 95 corroborates the above claim by simulation results. For the same data rate, hysteresis increases as we increase load capacitance. How

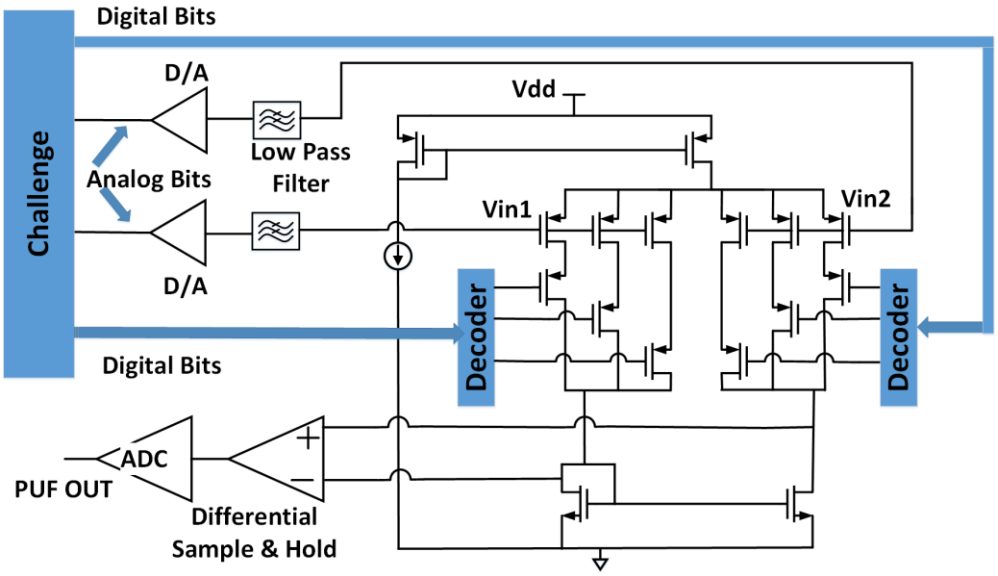


Figure 91: Modified PUF incorporating spatial randomness and memory effects (for architecture 2)

this memory effect enhances security of the PUF is explained in Figure 92. For a set of fixed digital input bits (a fixed differential structure is selected), a stream of "analog bits" are imparted on the analog PUF. Let's consider at time t0, for input x0, output is y0. At t=t1, for x=x1, output will make a transition from y0 to y1. If before completing this transition, at t=t2, input is changed to x=x2, then circuit tries to go to stable state corresponding to x=x2, if we keep on doing this and sample at t=tn, then sampled output yn is function of all previous inputs (see Eq. 98). For all previous PUFs, this sequence dependency were not exploited.

$$y_n = f(x_0, x_1, \dots \dots x_n) \tag{98}$$

In order to support the above argument an experiment is conducted where digital bits are kept fixed and two different sequences ($x_0 x_1 \dots . x_n \; and \; x_0' x_1' \dots . x_n'$) where end symbols ($x_n \; and \; x_n'$) are same. We get two different ADC codes at t=t1 and t=t2 (see Figure 92).

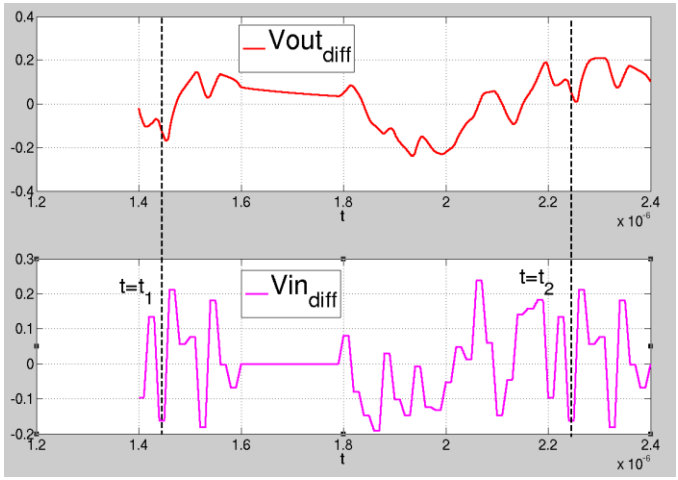


Figure 92: Output response v/s input challenge curve for analog PUF (showing memory effect)

In order to exhibit sequence dependency (memory effect) for architecture 1, we construct one experiment where no spatial randomness is exercised (i.e. digital bits to select the amplifier from amplifier array were kept fixed) only input (analog bits in Figure 93) to the amplifiers are varied so that ADC code 100 is reached from all other possible ADC codes. The result of the above described experiment in shown in Figure 94. It can clearly be seen that the sampled analog output values from the amplifier are varying at every sampling instant though the input value is fixed (ADC code 100). The results explained above, establishes our claim of memory effect.

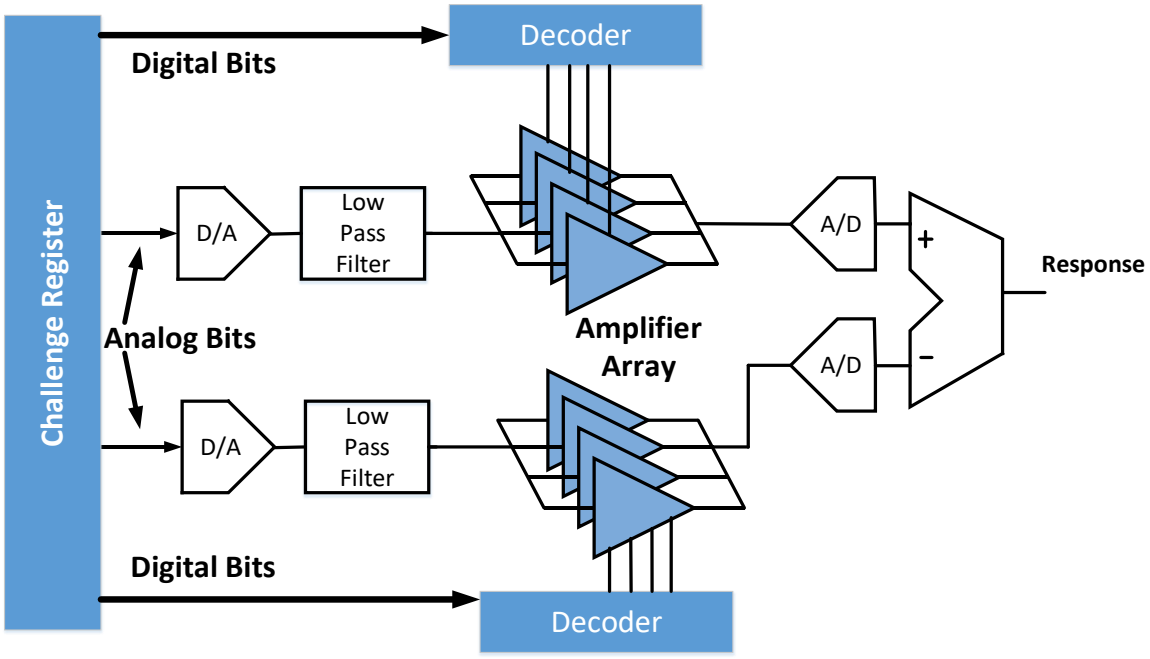


Figure 93: Modified PUF incorporating spatial randomness and memory effects (for architecture 1)

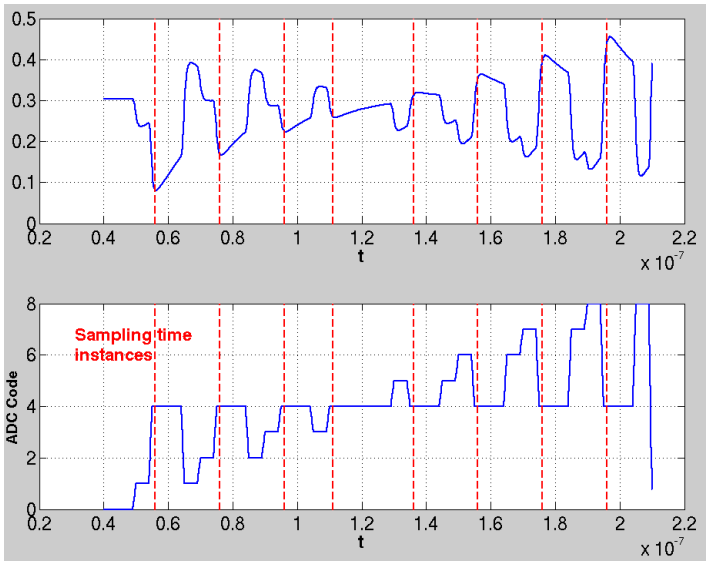


Figure 94: Output response v/s input challenge curve for analog PUF (showing memory effect)

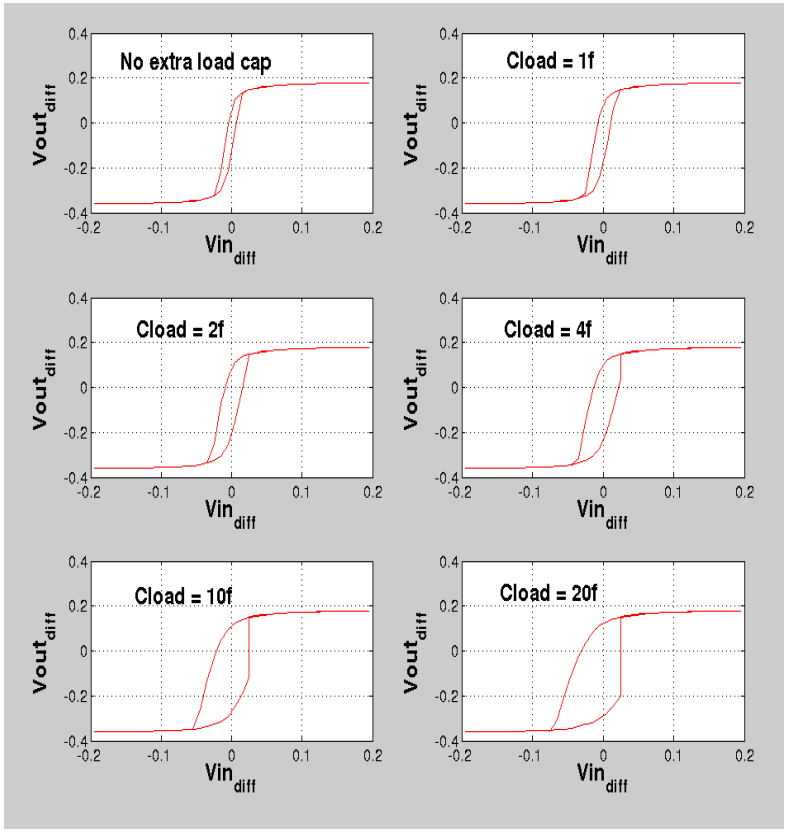


Figure 95: Extra load capacitance v/s hysteresis (memory) in transfer function of the differential amplifier

One plausible short coming of the proposed PUF in Figure 89, is saturation. As the amplifier is operated at subthreshold, it is very sensitive to threshold voltages change of the differential pair transistors (M1 and M2).A large change in v_{th} may drive the amplifier into saturation or close to saturation at either rail. Once the amplifier output is clipped to the positive or negative rail voltage, prior memory effects are lost. We conducted an experiment to find where the above said saturation occurs. We have seen that if $v_{th}(M_1)$ is varied +10% and $v_{th}(M_2)$ is varied -10% then the amplifier output is close to positive rail (see Figure 96) and memory effects are weaker than otherwise. It is very unlikely that every differential pair will have this huge mismatch in threshold voltages. Regardless, the architecture of Figure 91, overcomes this problem by exploiting both spatial variations and memory effects to make PUF reverse engineering difficult. By switching back and forth among various differential pair combinations, even weakly saturated amplifiers can be forced to exhibit sequential dependency.

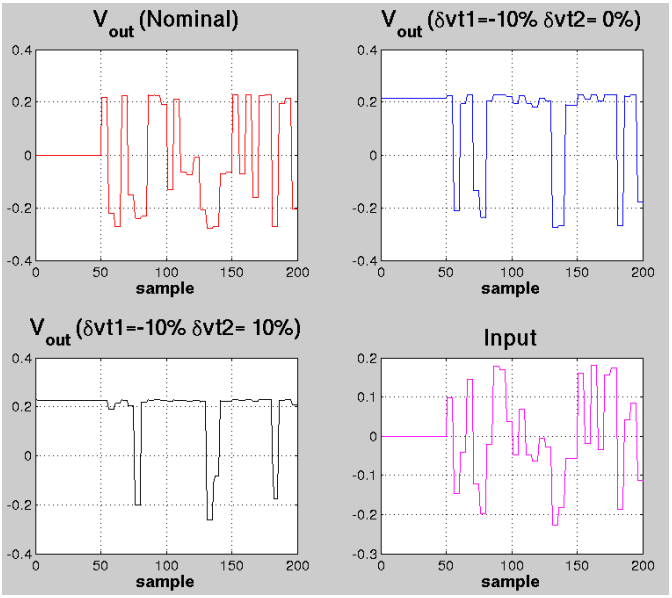


Figure 96: Differential amplifier output for various threshold voltage variation

6.4 Key Generation and IC Authentication Using PUF

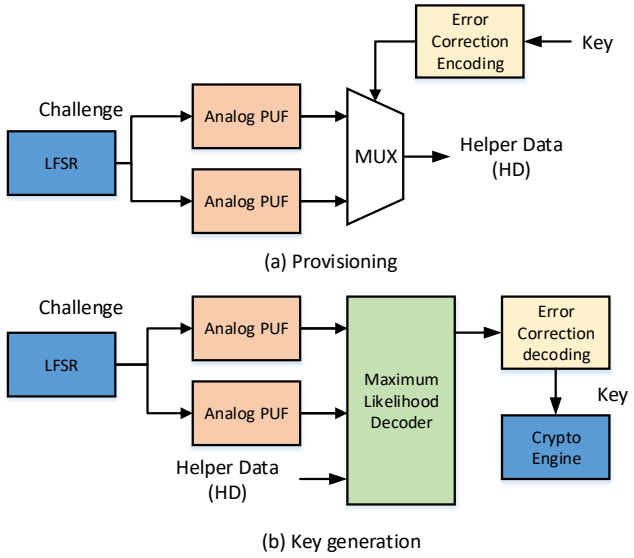


Figure 97: PUF in key generation

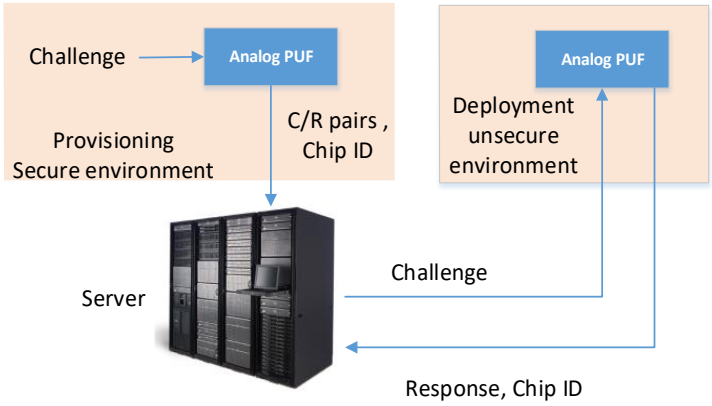


Figure 98: PUF in IC authentication

Key generation for AES cryptographic engine (or any such cryptographic requirement) using PUF is explained in Figure 97. During the provisioning stage (Figure 97a) the key is applied and corresponding to PUF response, a helper data is put out which hides the key (see Algorithm 6). The produced helper data is chip specific as PUF response

will by nature vary across chips. The challenge used for generating helper data is hardcoded (only a single challenge is used for key generation, so we only require week PUF) into the chip by a LFSR. After provisioning, helper data creation is permanently disabled by burning fuses, so an attacker can no longer apply and observe CRPs. In deployment, the chip specific helper data is applied (helper data is given to the user) to the chip and from the PUF response and the helper data, the key is regenerated and applied to cryptographic engine. In this work maximum likelihood decoder is used to retrieve the key from helper data and noisy PUF response. As shown in Figure 98, for IC authentication, during the provisioning stage a large number of CRPs of the chip are stored on the server. During deployment the server pings the chip with several challenges and the chip responds with chip ID as well as corresponding responses. Based on the maximum likelihood decoder (see Figure 99) the responses are decoded and matched with stored responses. The response of the analog PUF is string of 3 bit ADC codes; in the simulation section we have shown that maximum absolute code difference we can get from PUF responses due to environmental noise is 1 (see Figure 102). In key generation single bit of key is not encoded with one PUF symbol, rather it is encoded with B number of symbols for better reliable decision making in decoding. By simulation we have seen that B>4 is extremely reliable for key generation.

Algorithm 6: Key generation

```
Input: HD, PUF'_i      ( PUF'_i : noisy response of i^th PUF )
Output: Key
L: length of HD
B: #HD symbols required to encode/decode single key bit
Key = [ ]
For i=1:B: L
      M1=M2=0
      For j=0:(B-1)
            D1=abs(HD(i+j) – PUF1(i+j))
            D2= abs(HD(i+j) – PUF2(i+j))
            // M: matching count (see Figure 99 )
            M1+= (D1>1)?0:1
            M2+= (D2>1)?0:1
       Keybit=(M1>M2)?1:0
       Key=[Key keybit]
```

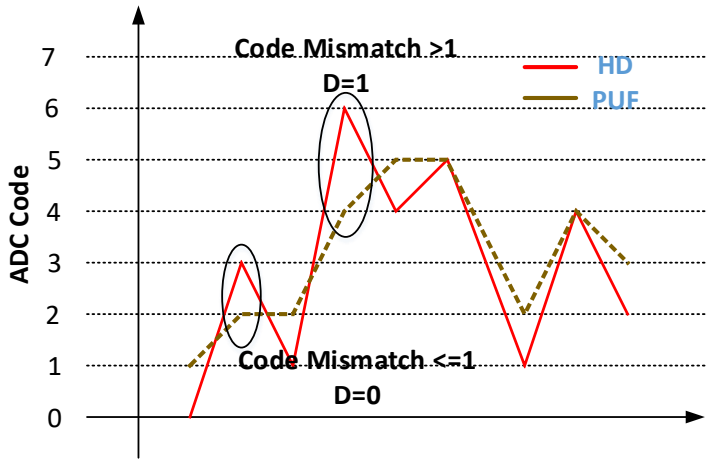


Figure 99: Maximal likelihood detection

6.5 Challenge Engineering

As it is explained in section 6.3, the PUF output response is input sequence dependent, and there are some sequences that are better that the others in terms of various PUF metrics (uniqueness, reliability etc.). Challenge engineering is to find a challenge

(sequence of input symbols) which is optimized for PUF metrics. For a weak PUF the challenge can be hardcoded into the PUF, and we need a challenge which will maximize uniqueness (every chip will have different helper data), and will have fewer unreliable response bits (increase reliability). Uniqueness and reliability of an analog PUF defined in [11] is followed in this work. A hierarchical clustering is used to maximally cluster the PUF responses (R_i) for a given challenge C, to enumerate and thereby quantify uniqueness (Eq. 99).

$$uniqueness = \frac{\max(Number\ of\ clusters\ of\ R_i)}{n} \tag{99}$$

Where R_i is response of i^{th} PUF device, n is total number of PUF devices used and $i = 1,2 \ldots\ldots n$. On the other hand reliability is a measure of reproducibility of PUF response at diverse temperature and voltage conditions. Reliability of a PUF is explained by Eq. 100,101 and 102.

$$x(j,l) = 1 \quad if\ |R(j,l) - R(j,l_{nominal})| > code\ margin \tag{100}$$
$$= 0 \text{ otherwise}$$

$$Sum_x = \frac{1}{mk}\sum_{j=1}^{m}\sum_{l=1}^{k} x(j,l) \tag{101}$$

$$reliability = (1 - Sum_x) * 100\% \tag{102}$$

Where m is total number of symbols, k is total number of environment corners. Based on above definitions of uniqueness and reliability, a genetic algorithm based challenge crafting algorithm is proposed (see Figure 100). For every m candidate stimuli, n PUF devices and k environment corner n*m*k responses are simulated. Every stimulus is assigned a weight W(i) (Eq. 103) and based on their weights stimuli are ranked. In every

iteration of genetic algorithm only elite stimuli are kept and others are discarded. New stimuli are created from elite pool of stimuli by genetic cross over and mutation.

$$W(i) = W_{uniqueness}(i) * W_{reliability}(i) \;\; i = 1,2 \ldots m \tag{103}$$

Where $W_{uniqueness}(i) = \sum_{j=1}^{k} uniqueness$ (sum of uniqueness at environment corners) and $W_{reliability}(i) = \sum_{j=1}^{n} reliability$ (sum of reliability of all PUF devices). For a strong PUF we start the Genetic algorithm from different initial population and run it multiple times to come up with many challenges. We run this algorithm on 1000 random process varied devices (+- 10 % vth and length variation) to tune the challenge bits of the PUF.

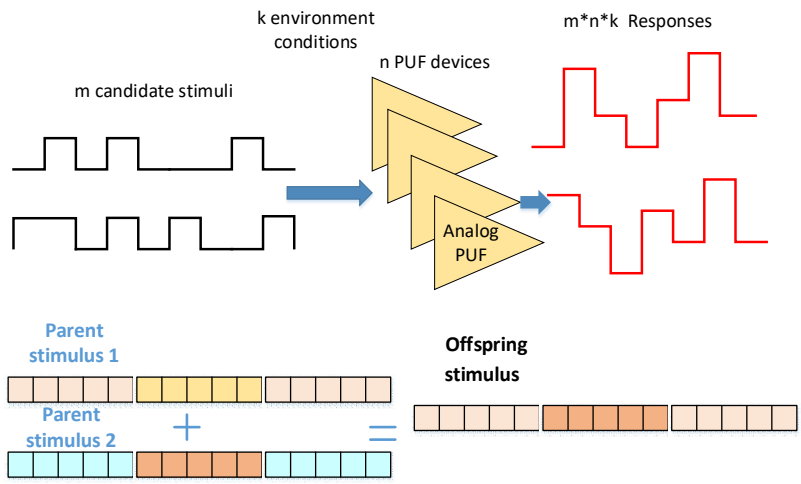


Figure 100: Challenge crafting

6.6 Experimental Results

6.6.1 PUF Performance Metrics

We cannot control voltage fluctuations and environment temperature condition of an IC in deployment. Because of the above mentioned environmental variations, PUF responses

may become noisy and may not match the helper data extracted during provisioning. Previous PUFs relied on error correction coding to tackle this issue. As we have mentioned, we will tolerate 1 code difference during decoding, thereby becoming less reliant on error correction coding. We conducted an experiment where 2000 process varied PUFs were taken and simulated at extreme voltage and process corners (VDD varied -+ 10% and temperature is varied from -20^{O}C to 120^{O}C), to check their reliabilities. Reliability of a PUF is defined as percentages of output bits that can be reproduced at extreme environmental condition. The result is shown in Figure 101 and Figure 102 respectively. It is apparent that amplifier biased at deep subthreshold (VDD=0.5) is less reliable than the amplifier biased just at the threshold (VDD=0.55) region of operation.

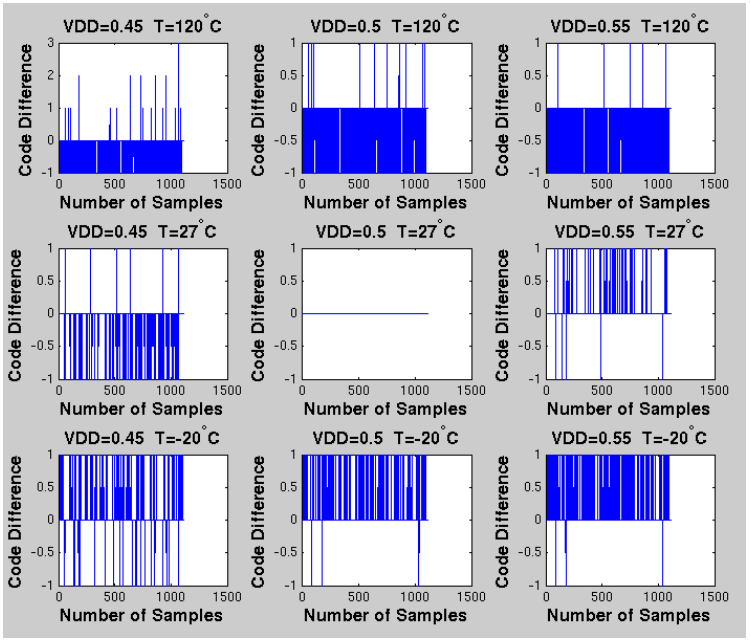


Figure 101: Reliability of PUF biased at deep subthreshold (architecture 2)

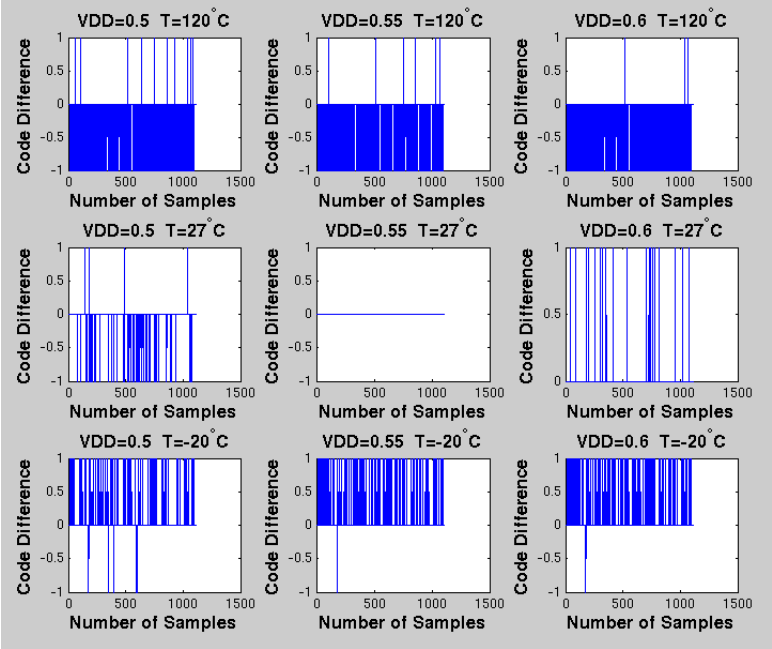


Figure 102: Reliability of PUF biased at just subthreshold (architecture 2)

From the same process statistics (+-10 % vth and length variation) 2000 arbiter and proposed analog PUFs were created. For a 512 bit input challenge, how many unique responses were created out of 2000 PUFs are compared and shown in Table 29. Result in Table 29 clearly shows that for a given number of bits, analog PUF effectively exhibits 2X more randomness than the arbiter PUF.

Table 29: Comparison of arbiter PUF and proposed analog PUF

	Arbiter PUF	Proposed Analog PUF	
		architecture 1	architecture 2
uniqueness	0.35(700/2000)	0.7(700/1000)	0.8(1600/2000)
reliability	99.6 %	99.0	99.5 %

6.6.2 Robustness to Security Attacks Using PUF Model Learning

In this section we show that the proposed PUF is difficult to model using black box experiments and therefore provides robustness to PUF clone based security attacks. In

Section 6.3, amplifier memory effect is discussed and it is seen that the PUF output is dependent on previously applied inputs also. A model for capturing the sequence dependence effect is proposed in Eq. 104. To find the model *f*, a neural network (see Figure 103) with tapped input delays to capture dynamic response of time series data is used. Matlab's trainlm function (Levenberg–Marquardt algorithm) is used to train the network.

$$y_n = f(x_n, x_{n-1}, \ldots . x_{n-k}) \quad where\ n-k>0 \tag{104}$$

$$y_n = a_0 + a_1 x_n + a_2 x_{n-1} + \ \ldots \ldots . + a_{n-k+1} x_{n-k} \tag{105}$$

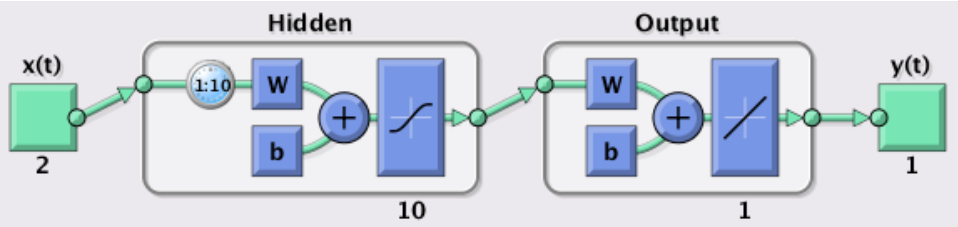


Figure 103: Neural network used for analog PUF model building

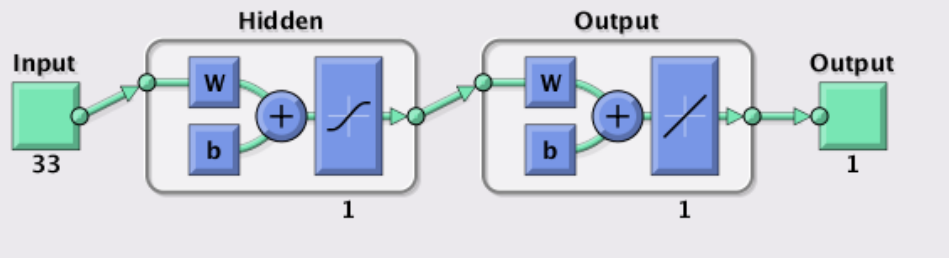


Figure 104: Neural network used to build model of Arbiter PUF

We have also used linear regression (Eq. 105) to predict this model behavior. For this experiment, 8 bit ADCs at the inputs of Figure 88 and 3 bit ADCs at the output are used. In [130] the authors have used logistic regression and evolutionary techniques to solve the model of Arbiter and XOR-Arbiter PUFs. Models of Arbiter and XOR-Arbiter PUFs as described in [130, 131] are given in Figure 105. Detailed derivation of delay

equations and parameters are given in [130, 131]. Eq. 106 and 107 describe top and bottom path delay models of an arbiter PUF.

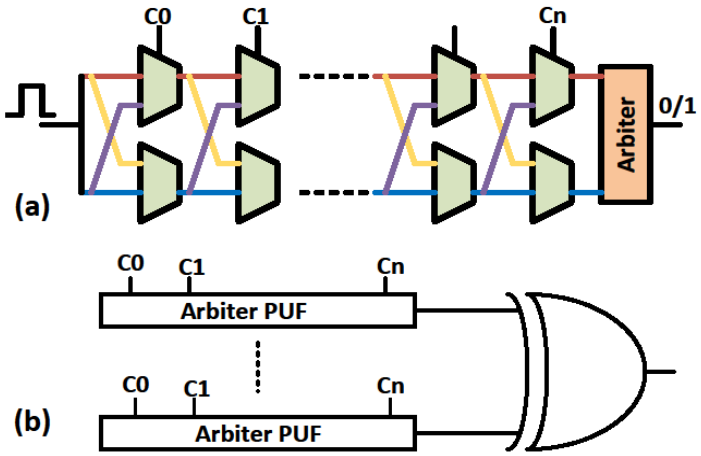


Figure 105: Arbiter PUF

$$d_{top} = \sum_{i=0}^{n} (-1)^{c_i} v_i + v_* + \sum_{i=0}^{n} (-1)^{p_i} y_i + y_* \quad (106)$$

$$d_{bottom} = \sum_{i=0}^{n} (-1)^{c_i} v_i + v_* - \sum_{i=0}^{n} (-1)^{p_i} y_i - y_* \quad (107)$$

$$where\ p_i = c_i xor c_{i+1} \dots \dots xor\ c_n \quad (108)$$

$$Arbiter\ Out = 1\ \ if\ d_{top} > d_{bottom} \quad (109)$$
$$= 0\ otherwise$$

$$XOR\ Arbiter\ Out = \bigotimes Arbiter\ Out_i \quad (110)$$

Linear model building has been used to reverse-engineer arbiter PUF and XOR arbiter PUF models. Neural networks that model arbiter and XOR arbiter PUFs are shown in Figure 104. For 32 bit arbiter PUF and XOR arbiter PUF, the number of unknown parameters to be estimated for linear model building are $n + 1$ and $(n + 1)^l$, respectively. The model building success rate is shown in Figure 106. It is apparent from Figure 106, that reverse engineering of analog PUF should be extremely difficult for an adversary. While matching PUF response the same definition used in maximum likelihood decoding

described in section 6.4 is used (i.e. one code difference will be considered a match).In [130] the authors have used logistic regression and evolutionary techniques to solve the model of Arbiter and XOR-Arbiter PUFs. Using the models of arbiter and XOR-arbiter PUFs as described in [130, 131], and using same process statistics as of analog PUF, we try to build models of arbiter and XOR-arbiter PUFs using neural network. To have an apple to apple comparison we have compared various performances of model building attacks on various PUFs w.r.t number of input bits. With 8000 input bits an attacker is almost 95% accurate to predict arbiter PUF response and prediction accuracy falls to 62% for XOR arbiter PUF. Prediction accuracy for proposed analog PUF is almost 33%.

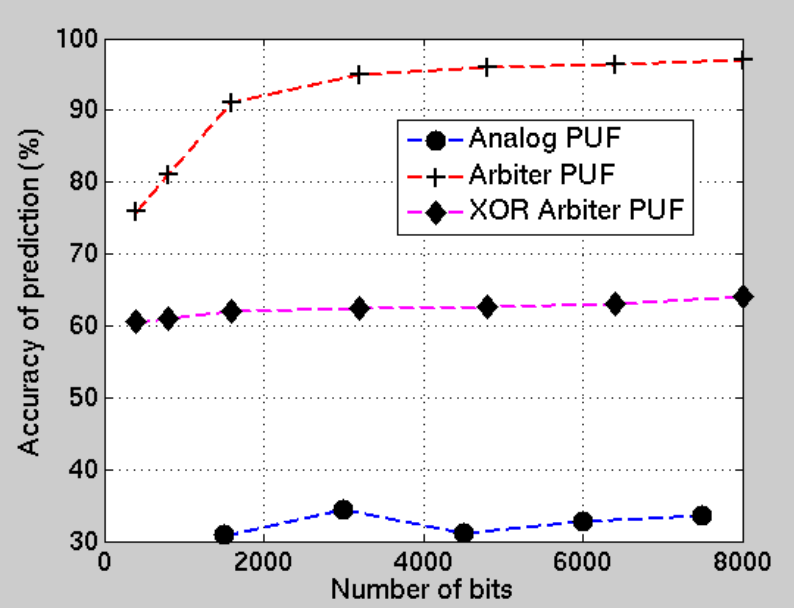


Figure 106: Model building attack on various PUFs

6.7 Analog PUF Performance Quantification Metric

In this section we will discuss how to compare and quantify the performance of any analog block to be used as PUF. To date we use uniqueness metric to compare two PUFs.

Uniqueness is manifestation of process variation present in the technology and how the underlying circuit is exploiting that process variation to create different signatures. It is required to stimulate the candidate PUFs (sampled from large number of process corners) with a large number of stimulus to measure uniqueness of a PUF. Uniqueness provides a measure of randomness harnessed from the device. In previous sections we have discussed how memory effects in analog circuit can be fused into special randomness to enhance the randomness harnessing capability of an analog circuit. Here we will discuss a metric (we named it generating matrix) to compare among analog circuits suitable for signature generation.

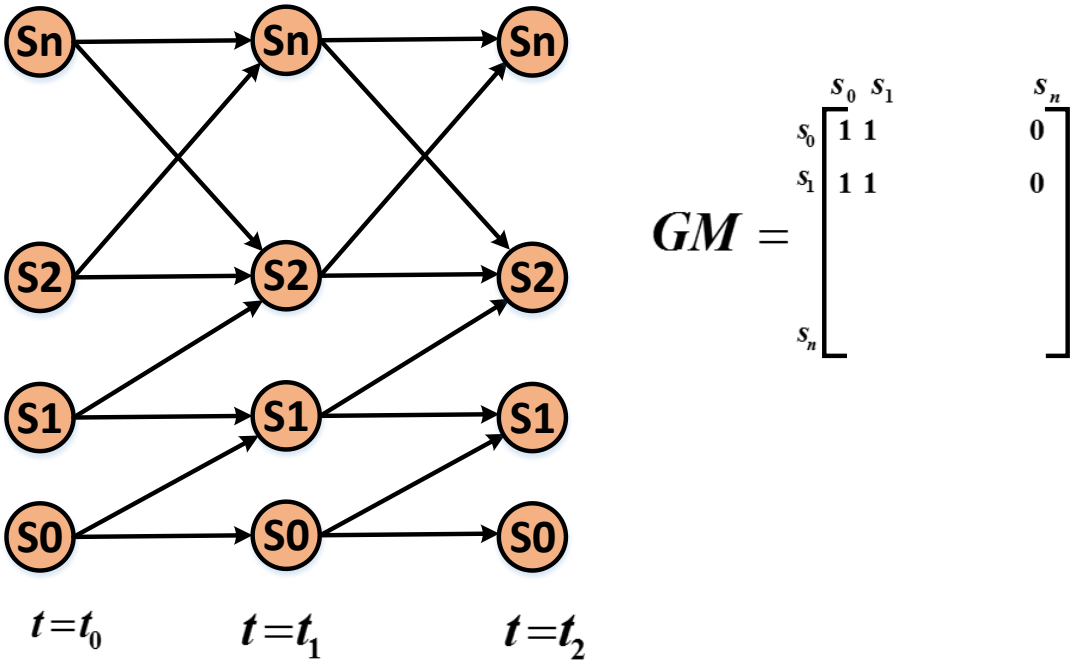


Figure 107: State transition graph and corresponding generating matrix of the PUF

Lets assume that the analog output is quantized into (n+1) number of levels. If we sample the analog output at regular δt interval then sinature of length m is given in Eq.

111. In Figure 107, we are showing a state transition graph where an arrow shows a possible transition between states. Signature of length m is a path in this state transition graph of level m ($t = t_0\ to\ t = t_m$). Higher the number of such paths possible, better will be the uniqueness of the PUF.

$$Signature = \{s_1^{i1}, s_2^{i2}, \ldots . s_m^{im}\} \quad (111)$$
$$where\ i1, i2, \ldots im\ are\ integer\ and\ can\ take\ any\ value\ between\ 0 \text{ and } n$$

Generating matrix is a connection matrix (nxn) of the state transition graph of two adjacent levles. Number of unique possible paths from the state transition graph is given by Eq. 112. If we assume that the generating matrix remains constant between any two successive levels then Eq. 112 becomes Eq. 113.

$$Number\ of\ unique\ paths = GM_{0,1} GM_{1,2} \ldots . GM_{m-1,m} \quad (112)$$
$$where\ GM_{i,j}\ is\ the\ generating\ matrix\ of\ the\ levels\ i\ and\ j$$
$$Number\ of\ unique\ paths = GM^m \quad (113)$$
$$where\ GM = GM_{0,1} = GM_{1,2} = \cdots = GM_{m-1,m}$$

Strength of generating matrix is defined as the number of ones in the GM matrix (given in Eq. 114). Higher the number of ones, more is the chance of producing an unique signature. Strength of generating matrix is formulated as a metric to compare two analog structures for building PUF.

$$Strength\ of\ Generating\ Matrix = \sum\sum GM(i,j) \quad (114)$$

Figure 108 is corroborating the idea explained above with simulation data. For n=7 (8 number of quantization levels) and m=5 (signature length) in Figure 108(a) we have shown how the number of unique signatures increases with generating matrix strength. In Figure 108(b) we have shown how the average code difference among the generated signatures are increasing with generating matrix strength.

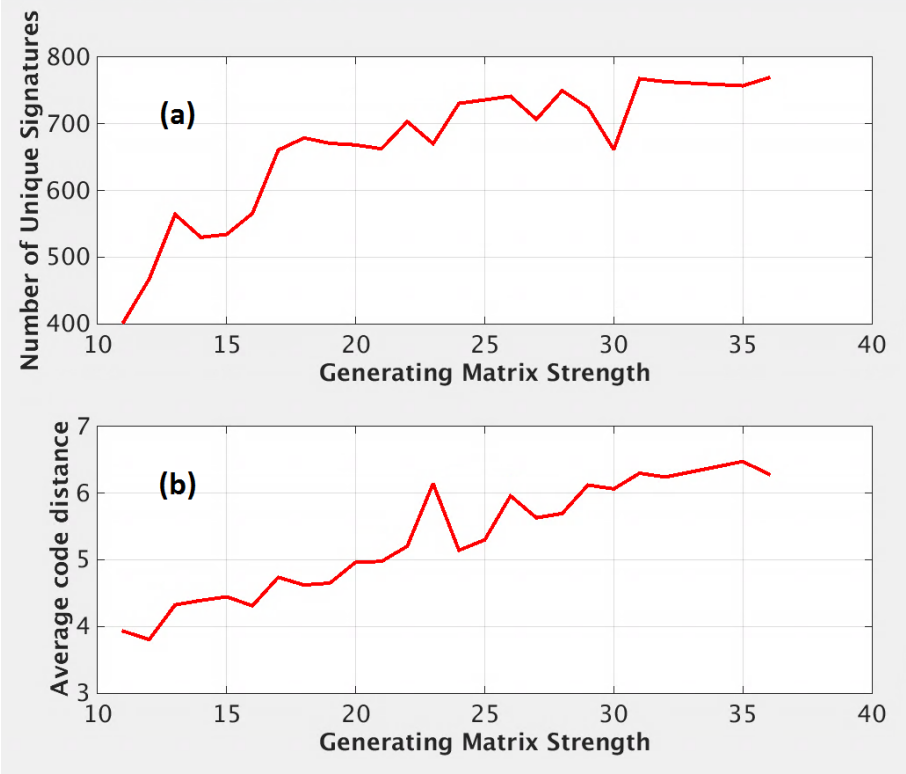


Figure 108: (a) Generating matrix strength vs number of unique signatures (b) Generating matrix strength vs average code distance

6.8 Applications to Public PUF

In this section we will discuss how the proposed analog PUFs can be leveraged to build public PUF. In conventional PUFs the challenge response (CR) pairs used to authenticate any IC are stored in a protected server. During provisioning stage these CR pairs for every manufactured ICs are generated and stored in the server in a trustworthy environment. In deployment when an IC is pinged with a set of challenges, it produces corresponding responses and send them back to the server along with its own serial chip

ID. The server authenticates the IC by comparing the stored responses with the received ones. Provision and deployment operation of a PUF is shown in Figure 98. In public PUF instead of storing all the responses in a secured server a model corresponding to each manufactured IC is publicly available. Thus public PUF does not entail large storage space and security of storage server. Public PUF is predicated on the fact that circuit response is way faster than any of its model simulation. How these models are generated in the provisioning phase is explained below.

6.8.1 Public PUF Model Generation and Validation

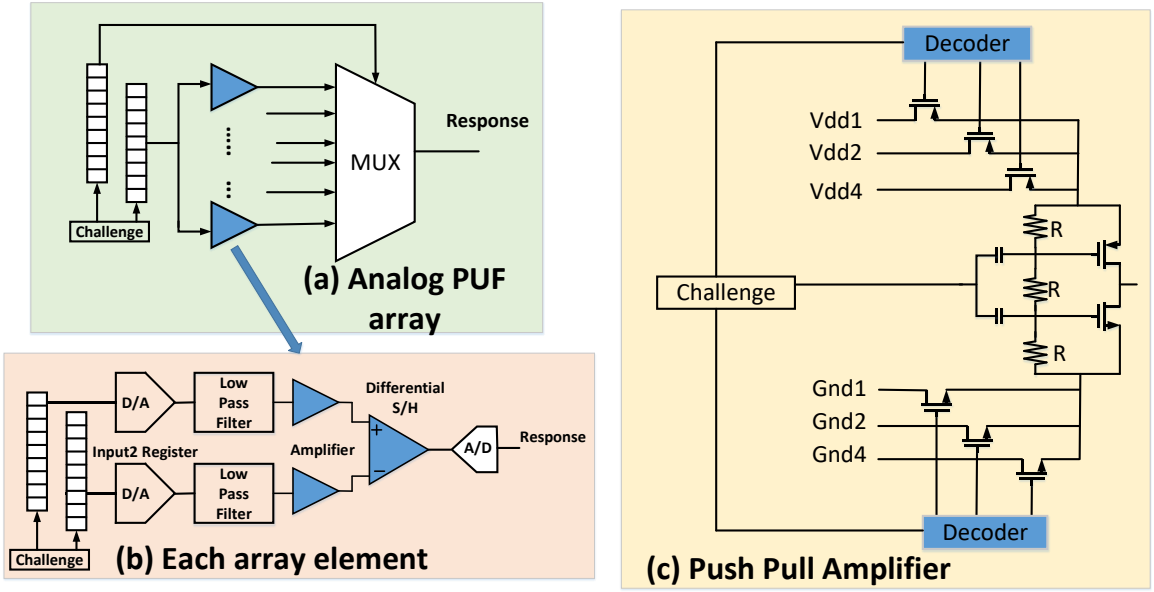


Figure 109: Public PUF architecture

Input and output of the amplifier (shown in Figure 109c) are observable during provisioning phase. After provisioning these ports are disabled so that the user (potential attacker) has no access to model building of the PUF. Every IC response deviates from its simulation response due to manufacturing process variation. The public model developed

for an IC in provisioning phase is a SPICE level model. As shown in Figure 110 we have formed an optimization problem with the objective of matching IC response to its model response. BSIM4 SPICE model of a transistor has more than 400 parameters in it. We keep few critical parameters (threshold voltage, oxide thickness, length, width etc.) as variables in the optimization while rest are kept at their nominal values to keep the optimization problem tractable. Error in fitness function of the genetic optimization is shown in Figure 111.

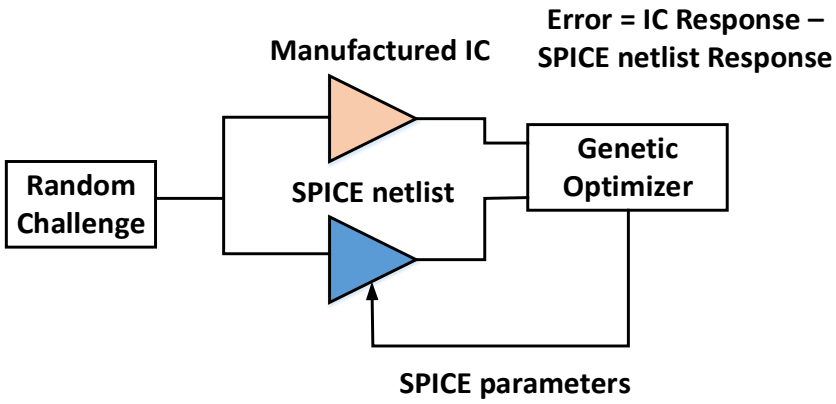


Figure 110: Public PUF model extraction

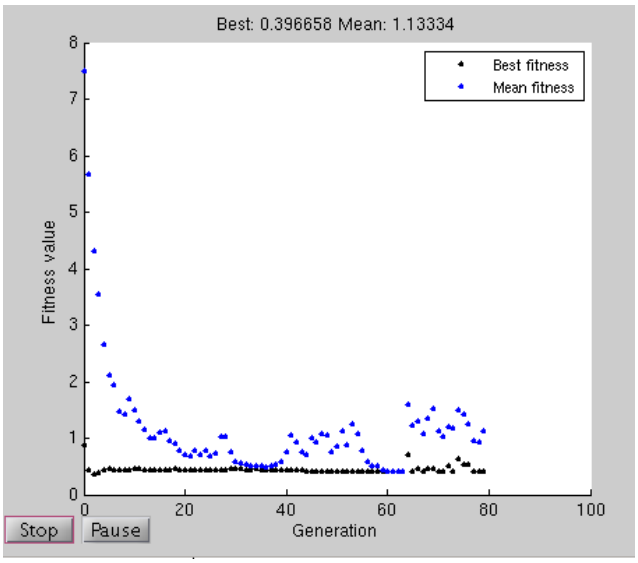


Figure 111: Public PUF model parameter optimization

Figure 112 is showing an original IC response and its extracted model simulation result for random input stimulus. Maximum analog response mismatch metrics are shown in Table 30. Error in symbol rate is 0.38% for 3 bit ADC. In Figure 113 we have shown symbol error rate v/s ADC bits for raw decoding and maximum likelihood (ML) decoding. In raw decoding with the increase of ADC bits, symbol error rate is increasing, while for ML decoding the error is fixed (very close to 0). The above experiment proves that the primary source of error in raw decoding is ADC quantization error.

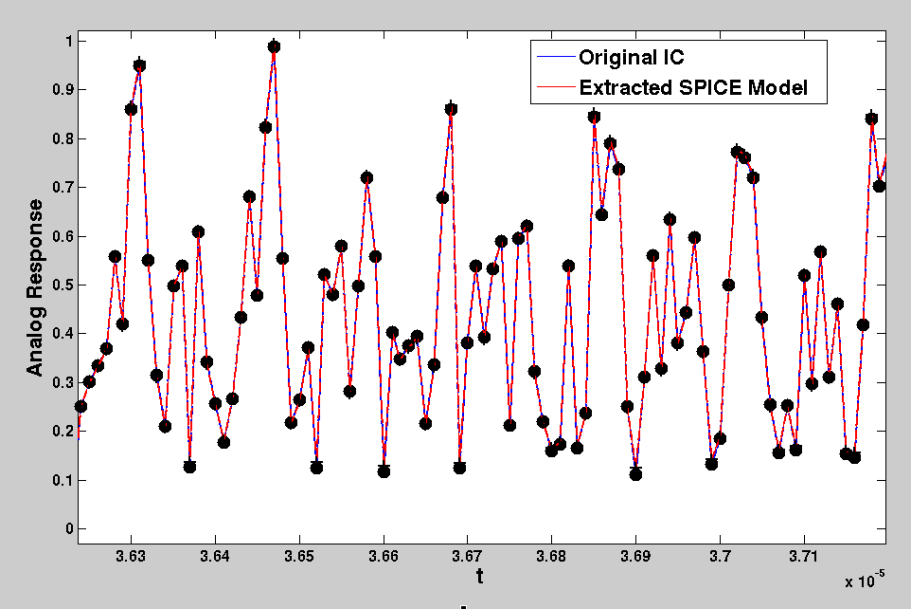


Figure 112: Public PUF model validation

Table 30: Public PUF model extraction validation

Maximum analog response mismatch	14mv
Average analog response mismatch	3mv
Number symbols transmitted	10000
error in symbols	38

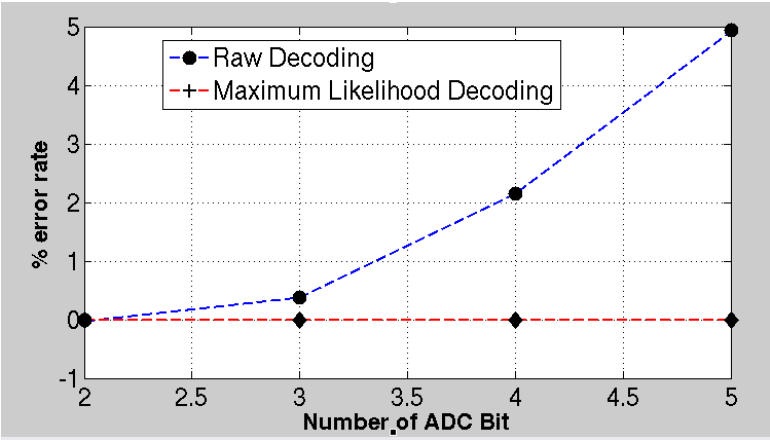


Figure 113: Extracted PUF model validation

6.8.2 Public PUF Architecture

The proposed PPUF architecture (shown in Figure 109) is an array of differential analog push pull amplifiers. The challenge bits are stored in memory. Challenge bits serve as input to these amplifiers (after digital to analog conversion and low pass filtering) and also in selecting various bias knobs of the amplifiers. Differential structure is employed to reduce the effect of ambient temperature and other circuit related noise. In every clock cycle they are used to select one of the element responses through a multiplexer. Multiplexer select lines are also coming from challenge bits. The structure shown in Figure 109 has no external feedback. Memory effect in response coming from analog amplifier. We can increase the effect of this hysteresis by employing an external feedback and memory elements (flip- flops). Figure 114 is showing the modified PPUF architecture. XORing external feedback and challenge bits (as shown in Figure 114) to control bias knobs of the analog circuit removes the possibility of parallel simulation for an attacker.

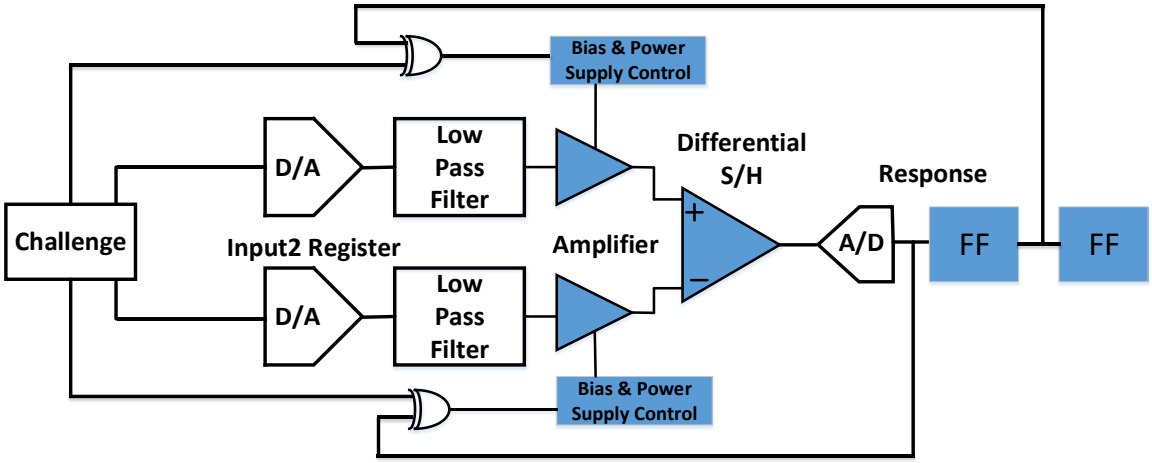


Figure 114: Modified PUF architecture

6.9 Conclusions

In this chapter we have demonstrated the potential of properly biased analog circuit in harnessing randomness from prevalent process variation in IC manufacturing [11]. The authors have also fused memory effects in analog circuit into spatial randomness due to process variation to thwart model building attacks on proposed PUF. Randomness harnessing capability of the proposed PUF, per input bit (or per unit given area) is far better than the state of the art PUFs. How the proposed analog PUFs can be used in public PUF has also been demonstrated. A metric (strength of generating matrix) is proposed in this work for better comparison of analog circuits to be used as PUF.

CHAPTER 7. CONCURRENT BUILT IN HIGH RESOLUTION PULSE PROPAGATION DRIVEN TROJAN DETECTION IN DIGITAL SYSTEMS

7.1 Introduction

Smart computing devices are ubiquitous now a days in human life. Integrated Circuits (ICs) are backbone of these computing devices. ICs have proliferated into every aspect of human life from smart phones, personal computers to medical devices, industrial controls automotive parts and military applications. Until recently, security of computation was mainly dealing with trustworthiness of softwares used in computation, and phishing attacks on softwares. Underlying hardware used was assumed to be trustworthy. The hardware cannot be trusted anymore. Defense Science Board of the US Department of Defense raises concern of hardware security in military critical equipments [132, 133]. IC manufacturing is getting very intense and complicated at sub nanometer technology nodes. IC manufacturing supply chain is now diversified. Logic design, verification, circuit synthesis, fab, packaging, testing all may be handled by various specialized teams. These specialized teams may be at different geographic locations and run by various organizations. Outsourcing of IC manufacturing has increased the possibility of intentionally tampering circuit at various possible design phases.

It is righty pointed out in [132] that trust has to be incorporated into an IC in design phase (Design for Security). There is a high need in industry for special mechanism built into the circuit to uncover these intentional malicious tampering. One way to uncover

hardware Trojan is to delayer the IC and take image of every layer with electron microscope and matching it with corresponding designed layout layer. Such an approach is very expensive and destructive. It also requires the use of appropriate (specially designed) CAD tools to detect inconsistencies between the actual physical design and the intended implementation.

Hence there is a need for nondestructive testing mechanisms for maliciously inserted Trojans into IC hardware. There has been research on detecting embedded Trojans by triggering it and comparing the logic values with the expected digital responses [134, 135]. The process of detecting Trojans by actually triggering it, is an expensive proposition. By design Trojan is triggered by a specific sequence of low activity logical events. Though Trojans rarely modify functionality of the design, their presence may alter other detectable characteristics of the IC behavior (side channel signature). Popular side channel analysis techniques for Trojan detection are (a) power measurement techniques [136, 137], (b) delay measurement methods [133, 138-141] . Other popular alternative Trojan detection techniques include Trojan activation time reduction, physical design based testing [142] and light emission technique[143]. Various state of the art hardware Trojan detection schemes and their drawbacks are as follows:

7.1.1 Reduced Trojan Activation Time

Hardware Trojans escape functional testing as Trojan activation time is too high. Presumably the Trojan payload circuit inputs are coming from very low activity factor nodes in the original logic circuit. In [2, 134, 135] authors have tried to artificially increase the activity factor of those nodes with few extra gates to catch anomalous activities in the

circuit. Even if k probable Trojan nodes in the circuit are identified, it is not known beforehand which combination of those will actually trigger the Trojan, and if it is sequential Trojan then the problem turns out to be more intense and complexity becomes many fold. For a combinational Trojan, all possible 2^k combinations needs to be checked. This makes Trojan detection extremely slow and infeasible. In this thesis chapter a technique is proposed to identify Trojans in less than k combinations for k probable Trojan nodes.

7.1.2 Power Measurement Techniques

In [136] the authors propose a Trojan detection scheme employing I_{DDQ} (steady state) current measurement at various sites across a chip. This method needs many Trojan infested and Trojan free chips for statistical training purposes to calibrate out process variation. Where the entire IC manufacturing is outsourced, it is impossible to obtain such Trojan-free chips in case they are being maliciously tampered with. In [137] the authors propose a power measurement scheme where Trojan activity is increased and original circuit activity is decreased in order to discern power consumption between Trojan affected and Trojan free circuits. This approach assumes that although the circuit activity is reduced, Trojan activity remains unaltered throughout the testing process. Key issues with this approach is that a single test vector has to be sustained for 25 clock cycles and process variation.

7.1.3 Path Delay Measurement Techniques

In [141] the authors have shown a path delay measurement based Trojan detection scheme where path delays of Trojan-free chips are collected and a chip signature is

constructed based on measured delays. In order to frame these signatures one needs to know apriori which chip is Trojan-infested and which one is Trojan-free. The authors have assumed the use of chip delayering and electron microscope imaging to obtain this information. In [133] Kumar et.al discuss Trojan detection using path delay measurement in scalable encryption algorithm (SEA) ASICs. Path delay measurement based Trojan detection becomes ineffective in presence of process variation, as delay variation due to process can mask increase in delay due to Trojan insertion. In [140] the authors have analyzed the effects of process variation on path delay based Trojan detection. The authors in [133, 141] have not considered process variation with due diligence in their respective approaches. Cha and Gupta [138, 139] have proposed a Trojan detection scheme in presence of process variations (systematic/global process variations) by incorporating on-chip test circuitry and through use of special calibration procedure. The authors assume that all the ICs are either Trojan infested or Trojan free. The proposed methodology requires testing of a large population of circuits to statistically predict the existence of embedded Trojans.

In this thesis chapter, a novel very high resolution pulse propagation driven hardware Trojan detection scheme is proposed. The rest of the chapter is organized as follows: Contributions of this work is described succinctly in section 7.2. Trojan threat model used in this work is explained in section 7.3. Theory of pulse propagation through logic gates is described in section 7.4. Pulse propagation driven Trojan detection ideas are described in section 7.5. Current detector, pulse generator and other related analog circuit operations are described in section 7.6. Section 7.7 describes how the proposed Trojan detection technique can be integrated into digital design. Simulation results to corroborate the

efficacy of the proposed Trojan detection technique are given in section 7.9. Conclusions are drawn in section 7.10.

7.2 Contributions of this work

We have shown in our previous work of [1, 144] that a pulse killing technique is capable of detecting any stray extra capacitance in any node with unprecedented accuracy. Pulse killing technique has shown 20-25X times diagnostic resolution than delay measurement (and frequency measurement), in presence of process variation. The above pulse killing technique is independent of logic depth in the circuit, while the other majority of the Trojan detection techniques (delay measurement [139, 141], frequency measurement [145], power supply current measurement [146, 147]) lose diagnostic resolution with the increase in number of gates in the design. The proposed pulse killing technique is self-referenced, it does not require any process calibration and no Trojan free manufactured ICs are required. In presence of process variation in today's state of the art technology nodes, detecting Trojan is extremely difficult even if golden reference (nominal design) is provided. Comparison with other state of the art Trojan detection techniques are shown in Table 31. Major accomplishments of the proposed Trojan detection scheme are briefly pointed out as follows:

(a) Reference free Trojan detection amid manufacturing process variation.

(b) Higher diagnostic resolution than state of the art techniques [139, 141]

(c) Trojan detection time is linear (in terms of number of susceptible Trojan nodes) in the proposed approach as opposed to exponential in state of the art Trojan detection schemes [4, 5] by increasing transitional probabilities of susceptible Trojan nodes.

(d) The proposed technique is suitable for detecting ultra-small Trojans.

(e) The circuitry needed (pulse generator and pulse detector) to enable the proposed pulse propagation driven Trojan detection scheme can be incorporated into existing JTAG and boundary scan infrastructure of digital pipeline stage designs.

Table 31: Trojan detection comparison with similar delay and frequency based techniques

	Process variation calibration	**Requirement of Trojan free and Trojan infested ICs.**	**Detecting ultra-small Trojans**
[138]	Require to calibrate out global process variation.	All the tested ICs either be Trojan free or Trojan infested. This is required for statistical hypothesis testing.	Not possible to detect ultra-small Trojan employing only 3 to 5 gates.
[141]	Not required	Require both Trojan free and Trojan infested ICs for model building.	Not possible to detect ultra-small Trojan employing only 3 to 5 gates.
[147]	Not required	No such assumption is made. A self-reference metric is enumerated based on multiple current sensor readings.	Not possible to detect ultra-small Trojan employing only 3 to 5gates.
This work	Not required	No such assumption is made.	Geared towards detecting ultra-small Trojans

7.3 Trojan Threat Model

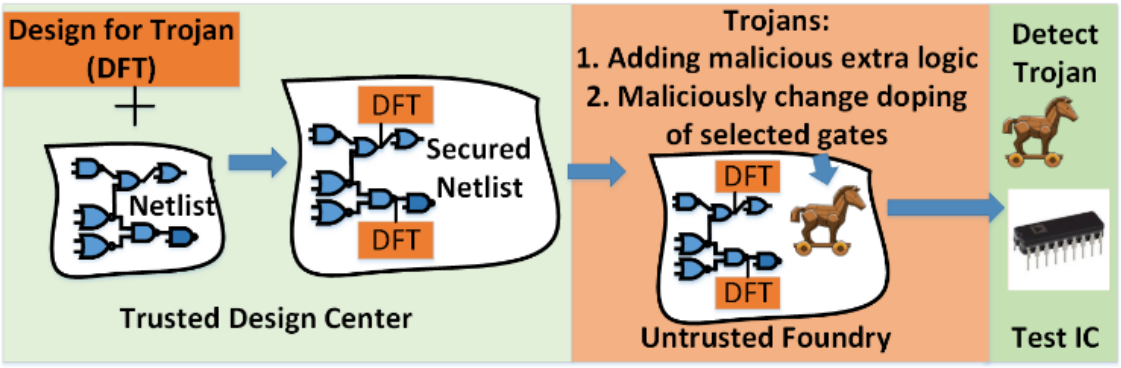


Figure 115: Trojan threat model

The threat model defines how a hardware Trojan is designed to attack and cause malfunction of intended operation or to leak secret information. In this work we have considered logic Trojan (putting extra covert circuitry to cause malfunction) and dopant Trojan (change in doping of some predetermined logic gate transistors to cause malfunction). Trojan attack model assumed in this work in IC manufacturing supply chain is shown in Figure 115. We have assumed that the design center is trusted. The designed netlist is secured by adding design for Trojan (DFT) analog IPs in the netlist. If any Trojan (extra logic circuitry, or dopant level Trojans) is incorporated in the foundry, it would be easily detected during IC manufacturing testing. This Trojan threat model is appropriate for all fabless semiconductor companies.

7.3.1 Extra Logic Circuity Trojan

As described in [148, 149] Trojan payload is the part of the circuit affected by the Trojan (part of the circuit where logic value is changed due to Trojan) and the act of causing it to have incorrect logic values is initiated by a Trojan trigger. Trojan can be combinatorial

or sequential. One example of each is given in Figure 116. Be it combinatorial or sequential the inputs to the Trojan trigger circuit is coming from original circuit nodes (low transitional probability nodes). When an original circuit node is tapped for Trojan trigger, it experiences an extra capacitance. Least capacitive tapping would be to use a minimum size inverter. Even when the Trojan is not activated this capacitance would be there due to loading of an extra gate (see Figure 117). It is this extra loading that we aim to detect with unprecedented high resolution using current sensing of pulse propagation through logic gates. Gate capacitance of a minimum sized inverter adds 0.2-1 fF capacitance to the tapped signal node (45nm PTM [150]). It will be shown in subsequent sections that in presence of 10% random process variation 880 aF of extra capacitance can be detected by the proposed pulse propagation technique.

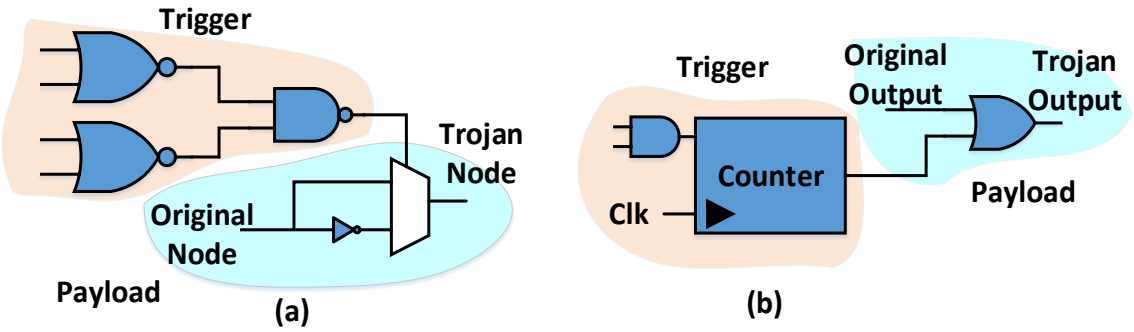


Figure 116: Trojan Models (a) combinatorial Trojan (b) sequential Trojan

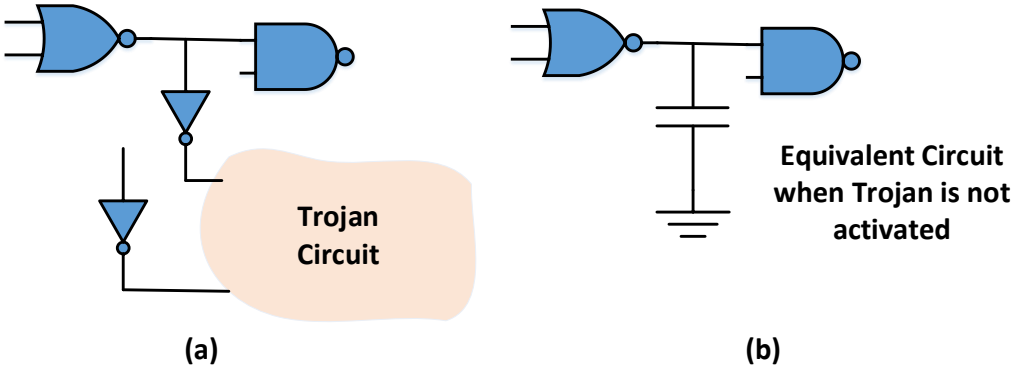


Figure 117: Tapping original circuit node for Trojan inputs (b) Corresponding equivalent circuit when Trojan is not activated

7.3.2 Dopant Level Trojan

In [151] the authors have shown that without even putting any extra circuitry, by modifying the dopant profile, Trojans can also be created in a circuit. These Trojans act as stuck at faults (short VDD/Gnd to output node). One such example is shown in Figure 118 where output of one inverter is stuck at VDD. Similar stuck at Gnd is also possible through NMOS transistor doping profile changing.

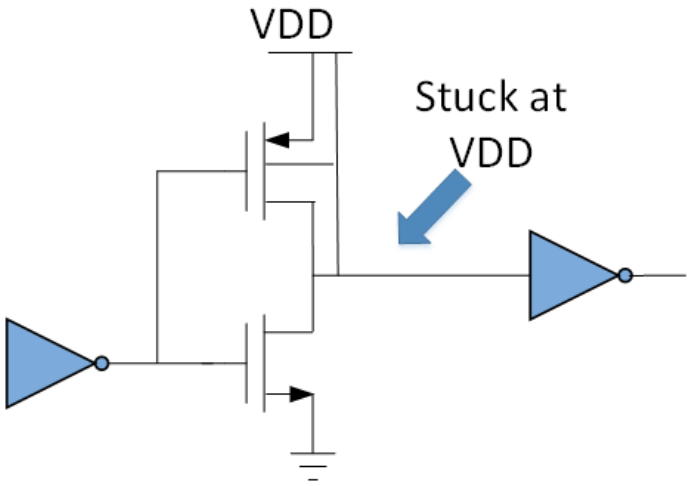


Figure 118: Dopant Trojan

7.4 Theory of Pulse Propagation Through Logic Gates

7.4.1 Pulse Killing

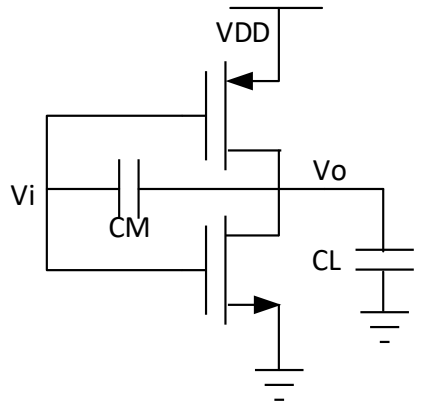


Figure 119 : An inverter

Without any loss of generality, pulse propagation through an inverter chain is analyzed here. The same concept is applicable to a chain of diverse types of CMOS logic gates. Forming a closed form equation describing input output relationship of an inverter is subject to reasonable assumptions. J.R Burns was the first person to derive such equations for a step input. Kayssi et.al [152] have shown such analytical relationships for inverters under predefined input shape assumptions (ramp and exponential). Bisdounis et.al [153] have shown such relationship for submicron inverters. Solving Kirchhoff's current law at V_O (Figure 119) yields Eq. 115 and 116.

$$CL\frac{dV_o}{dt} - I_p + I_n - CM\frac{d(V_i - V_o)}{dt} = 0 \quad (115)$$

$$\frac{dV_o}{dt} = \frac{CM}{CL + CM}\frac{dV_i}{dt} + \frac{I_p - I_n}{CL + CM} \quad (116)$$

The equations can be normalized so that they are independent of technology parameters (technology parameters are used in normalization factor). This normalization

helps in intuitive understanding of their function and the formulation of generalized analytical equations. The normalization factors are given below in Eq. 117 and 118.

$$v_i = \frac{V_i}{V_{DD}} \quad v_o = \frac{V_o}{V_{DD}} \quad v_{tn} = \frac{V_{tn}}{V_{DD}} \quad v_{tp} = \frac{|V_{tp}|}{V_{DD}} \quad \beta = \frac{K_n}{K_p} \tag{117}$$

$$i_n = \frac{I_n}{K_n V_{DD}^2} \quad i_p = \frac{I_p}{K_p V_{DD}^2} \quad cl = \frac{CL}{K_n V_{DD}} \quad cm = \frac{CM}{K_n V_{DD}} \tag{118}$$

Using these normalization factors, the system of equations describing the input output relationship of the inverter is given in Eq. 119. Expressions for i_p and i_n can be found in [154].

$$\frac{dv_o}{dt} = \frac{cm}{cl + cm}\frac{dv_i}{dt} + \frac{i_p - i_n}{cl + cm} \tag{119}$$

Table 32: Measured technology parameters

CL	CM	β	VDD	v_{tn}	v_{tp}	K_n
0.22fF	0.06fF	0.8	1.2	0.39	0.5	105μA/V^2

Table 33: Minimum pulse width required for propagation through the inverter chain

tr/tf of Input Pulse	τ_n	τ_p	Min Pulse Width (SPICE Simulation)	Min Pulse Width (Numerical Solution)
1ps	15ps	9ps	18ps	18ps
6ps	15ps	9ps	13ps	12ps

There has been research on pulse propagation through logic gates for Single Event Transient (SET) analysis [154, 155]. In SET analysis, the objective is to determine if a SET pulse can propagate through a logic chain and cause a logic failure. This analysis is leveraged for the pulse propagation technique proposed in this work. The above system of

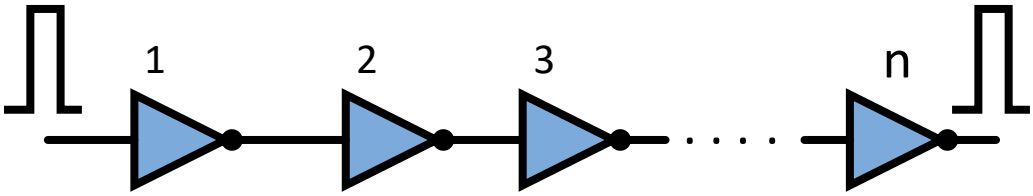


Figure 120. Inverter Chain

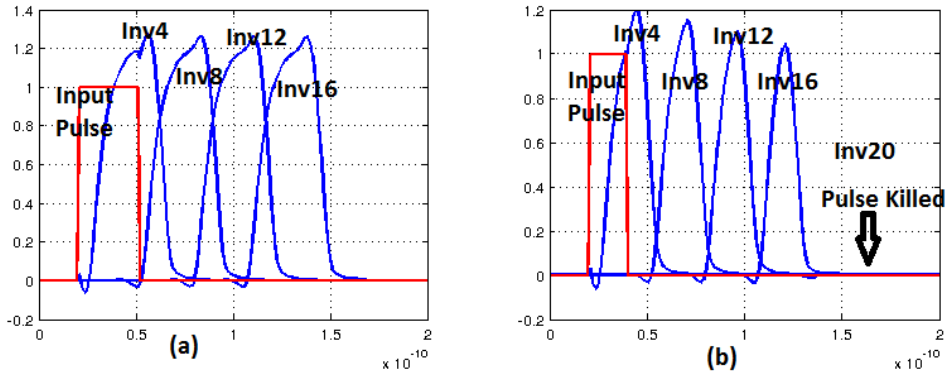


Figure 121. Pulse propagation (a) Input pulse width greater than the required minimum width (b) input pulse width less than the minimum required pulse width

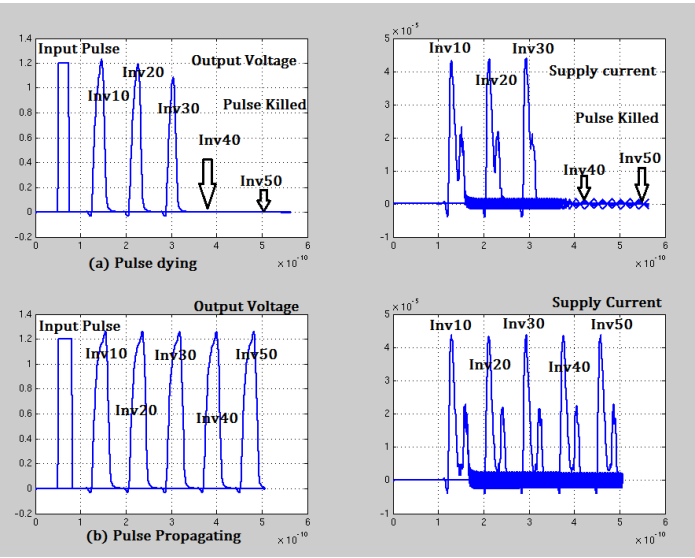


Figure 122. Pulse propagation (a) Input pulse width less than the required minimum width (b) input pulse width greater than the minimum required pulse width

equations are solved numerically and compared with SPICE level simulation results to

determine the effectiveness of pulse propagation in this work. In the following, all SPICE simulations use NCSU 45nm predictive technology. For inverter chain simulation, minimum sized inverters (ln=50nm lp=50nm Wn=100nm Wp=160nm) consisting of low threshold transistors (NMOS VTL and PMOS VTL) are used. The corresponding technology parameters are shown in Table 1. To propagate a pulse through a long chain of identical gates, at every gate, the transition swing should be rail to rail i.e. a positive pulse should reach VDD and a negative pulse should reach ground potential in order to continue the pulse propagation. If at any intermediate gate, rail to rail swing is not achieved at the gate outputs, the pulse will attenuate as it progresses through the chain (Figure 120) and eventually "die". If the input pulse rise/fall time (t_r/t_f) is smaller than the (10 to 90%) rise/fall (τ_n /τ_p) time of a gate then the minimum pulse width required to propagate the pulse through the chain of gates is $\tau_n + \tau_p$ [154]. It is found both by numerical solution and SPICE simulation that this is a greatly relaxed constraint for 45nm technology nodes and below. If $t_r(t_f) \ll \tau_p(\tau_n)$ then the constraint is even more relaxed. Table 2 above, depicts two such examples that illustrate this point. Figure 121 is showing two examples a) pulse width is above the required pulse width and pulse is propagating through infinitely long logic chain of inverters b) pulse width is less than the required width and gradual pulse killing. We have mentioned earlier that a presence of pulse can be detected by sensing supply current of a logic gate. When a pulse is propagating (dying), peak pulse voltage versus supply current at corresponding inverters are shown in Figure 122. Both peak current and rms current show almost linear relationship with peak voltage, Figure 123 corroborate our idea of detecting a pulse by sensing supply current of a logic gate.

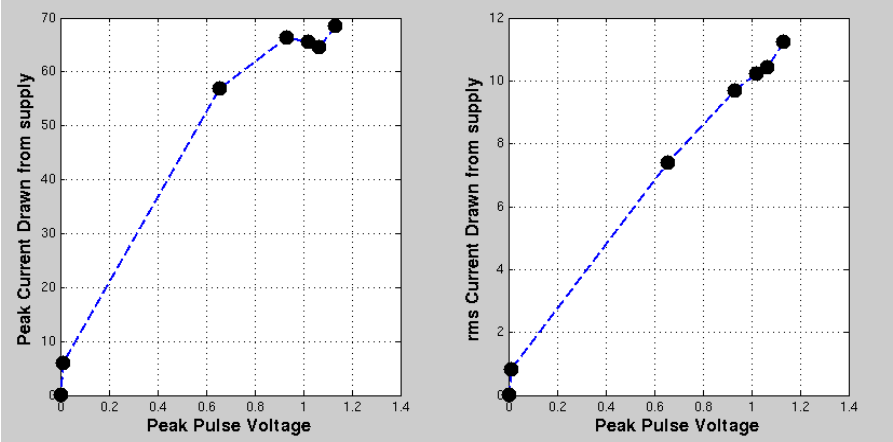


Figure 123: Peak pulse voltage vs current drawn from power supply

7.5 Approach: Pulse Propagation Driven Trojan Detection

Pulse propagation driven Trojan detection is a self-reference Trojan detection technique where no process calibration and no reference Trojan free manufactured ICs are required. It is assumed that the scan flip-flops drive the inputs of the logic circuit and outputs of the logic circuit are scanned out.

7.5.1 Pulse Sensitization

Firstly low transitional probability nodes are identified by stimulating netlist with sufficiently large number of random input stimuli. These are the probable Trojan tapping nodes (Trojan circuit will take input from these nodes). Pulse sensitization is performed from selected logic circuit inputs to appropriate logic circuit outputs in such a way that all the required circuit nodes are included in at least one sensitized path. The sensitization vectors are determined via a test generation algorithm (see Table 34) and we assume that all paths selected are single-path sensitized. The widths of the pulses launched are carefully minimized according to the discussion of Section 3, so that they pass unattenuated through the slowest gate in the path including worst case delay variation effects (e.g. if the slowest

gate in the path has nominal delay d and x% worst case delay variation is expected, then the minimum pulse width that passes unattenuated through an identical variation-affected gate with delay (1+(x/100))*d is computed). By induction (and confirmed via simulation), it can be shown that such a minimum sized pulse will pass from circuit input to output irrespective of the number of gates in the selected logic path. Note that if two gates in a sensitized logic path have different delays d1 and d2, with d2 > d1 and if the minimum pulse width is selected that passes through the gate with delay d2, then any additional load capacitance on the gate with delay d2 due to a Trojan will be detected as it will attenuate the pulse and "kill" it. However, a Trojan will be able to load the output node of the gate with delay d1 with a minimum capacitance CTrojan before it is detected by our pulse propagation procedure. Hence, for every test and for every node in the corresponding sensitized path, there is a minimum resolution CTrojan with which Trojans can be detected for that node corresponding to the applied test. If a logic gate is common to two or more pulse propagation tests, then the maximum resolution with which a Trojan can be detected at the output node of that gate is the minimum of the resolutions corresponding to each of the applied tests. This resolution is a function of the delay variation of a single gate as opposed to a logic path and thereby becomes independent of the number of gates in the path. In contrast, for path delay based Trojan detection, if there are N gates in a path and the max delay variation in each gate is x%, then the extra delay from a nominal delay path is N*x time units. This is the delay guard band that must be allocated to the path delay measurement in excess of which the presence of a Trojan is indicated. Consequently the value of CTrojan is significantly larger (25X) as that much capacitance needs to be attached to an internal node of the path before its worst case delay value is violated.

A key point to note is that logical effort based device sizing schemes that optimize a circuit for performance tend to equalize delays across all logic gates in a path irrespective of fanout. If all gate delay values are equalized asymptotically to a common value *d* and if there is *x%* worst case variation in delay at each node, *then* C_{Trojan} is the extra capacitance that causes the delay value of the gate to increase from d to (1+(x/100))*d and is the same for each node along the path, irrespective of its logic depth.

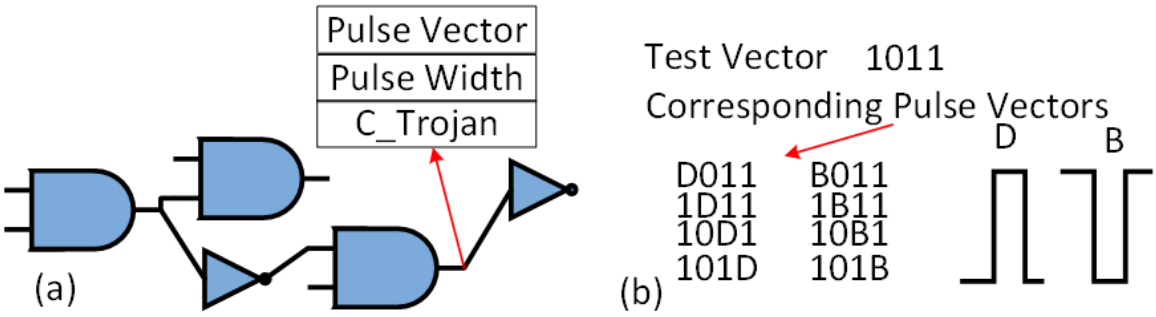


Figure 124: (a) Data structure for circuit node (b) Pulse vector

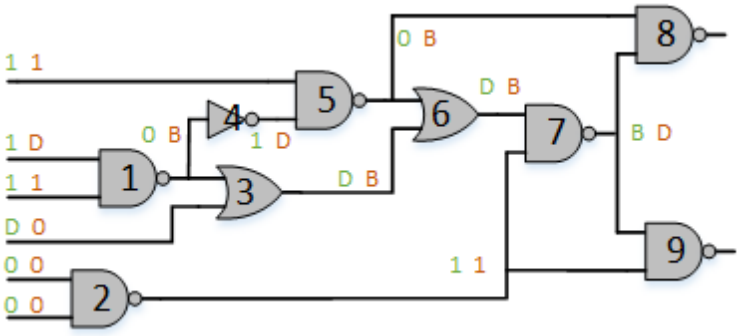


Figure 125: An example showing pulse test generation

For proof of concept, a simple D-Alg[156] based test generation algorithm for single path sensitization is given in Table 34. A simple example of pulse test generation for output node of gate 7 is shown in Figure 125**.** Two vectors 111D00 and 1D1000 are found by the D-algorithm. For pulse test vector 1D1000 both the inputs of gate 6 were carrying a pulse.

So this vector is dropped from the test vector set. For vector 111D00 only one input of a gate is carrying the pulse.

Table 34: Test generation for pulse based Trojan detection (voltage sensing)

/* generate data structure*/ [step 1] For each circuit node a data structure is created. (See Figure 124a) **/* generate test vector and corresponding pulse vectors*/ [step 2]** For a given node generate test vectors for s_a_1 & s_a_0 faults using D-alg /* for current sensing, the given node is considered output node as pulse detection point will be found later*/ Generate corresponding pulse vectors (see Figure 124b) **/*find sensitized paths*/ [step 3]** Apply a pulse vector and circuit response is captured. (Logic simulation) If any pulse (D or B) is found at the output, that pulse is traced back to the input. A traced path is valid if all the gates in the path carrying the pulse through only one of its input. (for example if both the inputs of a nand gate is D, output would be B ,it is carrying the pulse through both of its inputs and the path containing this gate is dropped from sensitized path list) **/*find minimum pulse width*/ [step 4]** The worst case process variation SPICE Netlist is taken and the pulse vector is applied. Starting with the minimum pulse width, the digital control of the pulse generator is increased until the pulse is detected at the output. This is the minimum pulse width required to propagate through the sensitized path corresponding to the pulse vector. **/*find minimum Trojan capacitance*/ [step 5]** The best case process variation SPICE Netlist is taken and a C_{Trojan} (starts with minimum 0.2fF) is inserted at a node in the sensitized path. The pulse vector is applied and C_{Trojan} value is increased until the pulse gets killed. This is the minimum capacitance detected by the pulse vector at that node. **Repeat steps 3 to 5 for all the pulse vectors [step 6]** **Repeat steps 1 to 6 for another given circuit node[step 7]**

5.2 Pulse Sensitization and Current Sensing Location Finding

The voltage based pulse detection requires logical effort based circuit sizing to keep each stage delays equal. In synthesis it is not always possible to keep all stage delays equal. The above mention technique does not take into account fan out problem in pulse propagation. If a pulse is so chosen (pulse width) that it would go unaffected through

maximum k fan out nodes, and in the path if it encounters any higher fan out node (>k) then the pulse will get killed. At higher fan out node, pulse experience an extra capacitive load that kills the pulse. So inside the circuit if we are testing presence of Trojan at a specific node of fan out k, then pulse cannot be applied from input through a path where fan out is greater than k at any one node. Similarly the voltage based technique aims to detect the pulse at the output scan flop, which is also not possible in presence of high fan out in logic circuit. And all the scan flops would require pulse capturing capability. In this work a single current detector is sufficient to detect presence of Trojans in 100's of paths. One such example is shown in Figure 126, where nodes m and n are Trojan probable nodes.

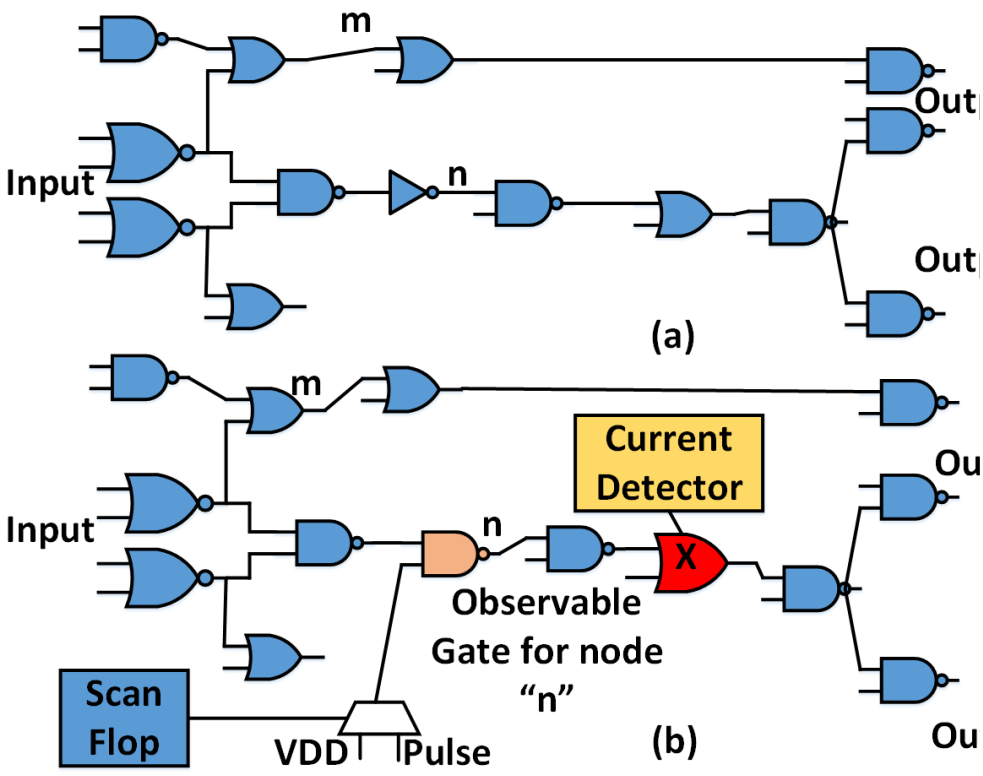


Figure 126: Finding current observation point and pulse application point for a given node (a) original circuit (b) modified circuit to incorporate Trojan detection DFS

A pulse can be applied to node m from primary input but cannot be applied to node n as there are no path from primary input to node n with max fan out of 1. For such nodes we have used one extra scan flop in scan chain (as shown in Figure 126) similar to the approach used in [2]. From node n there is no path to take this pulse at the output, on the basis of above mentioned fan out issue. If we observe supply current of gate "X" then we would be able to detect presence (or absence) of pulse at node n.

We have shown in previous section and will further show in experimental result section that under extreme process variation (+- 20 % random Vt change in ss, ff, nominal process corners) the proposed current detection technique can detect presence of a pulse in any one of 100 observation gates if these is a pulse propagation through any one of them (see Figure 134). For stimulus generation and path activation we have used a PODEM[157] like algorithm. We do not kill PODEM after finding one stimulus and one activation path, we keep on running it to get all possible stimulus and corresponding activation paths. For finding pulse application point and current observation point (see Table 35 and Table 36) we choose the best path and corresponding stimulus from PODEM output.

Table 35: Algorithm for implementing pulse detection based DFS (Design for security)

```
1.  findTransitionPrb(netlist, random stimuli)
2.  /*Stimulate netlist with sufficiently large number of random stimuli to find
    low transition probability nodes in the circuit*/
3.  S = {n_i| TrPrb(n_i) < Threshold Probability}
4.  //select nodes below threshold  transitional probability
5.  OG=null   // null observable gate set
6.  For each n_i ∈ S
7.      [stimulus_i, activation path]
          =FindStimulusAndActivationPath(netlist, n_i )
8.  // see algorithm described in Table 34
9.      If( activation path ==NULL)
10.         Add an extra scan flop (see  Figure 126)
11.         [stimulus_i, activation path]=FindStimulusAnd
12.         ActivationPath(netlist, n_i )
13.     og_i =FindobservableGate(netilts , n_i, stimulus_i )
14.     OG = {OG} U og_i
15. Assign VDD_currentDetector to gates {OG}
16. Assign VDD to all other gates
17. Synthesize design
```

Table 36: Algorithm of finding observable gate for a corresponding low transition probability node

```
1.  FindobservableGate(netlist, n, stimulus )
2.     {node values}=circuitSimulator(netlist, stimulus)
3.     f= fan out of node n
4.     {P}= Traverse netlist graph from node n towards
5.     output and track pulse till it reaches output or a
6.     higher fan out (>f) node
7.     //netlist is a unidirectional graph where logic gates
        are  nodes and connections are links
8.     P_longest =For all path listed in {P} find the longest path
9.     og= last gate of  P_longest
10.    Return(og)
```

7.6 Required Analog Circuits for On Chip Pulse Generation and Detection

7.6.1 Voltage Pulse Detector

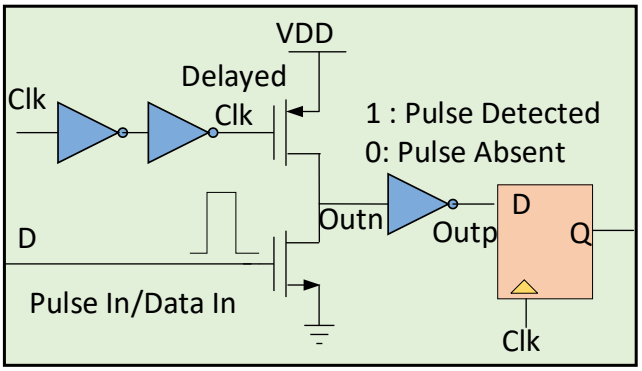


Figure 127: Modified latch with pulse detector

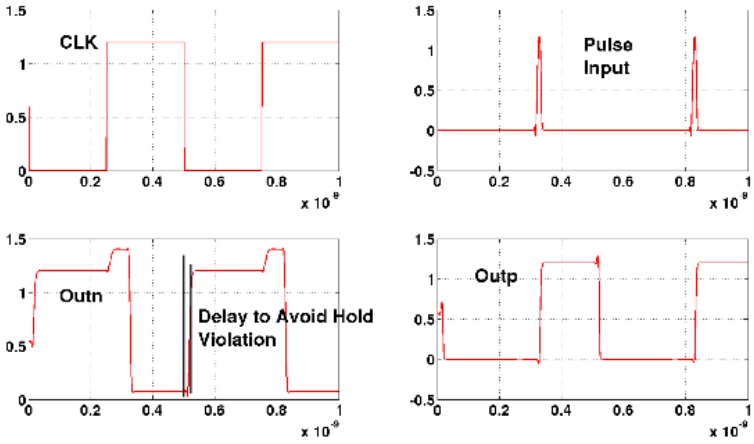


Figure 128: Pulse detector simulation result

The voltage pulse detector circuit is a carefully sized clocked inverter and is integrated inside the scan flip-flop circuitry. The node Outn (Figure 127) is pre-charged to VDD during the low clock phase. During the high clock phase if the input D does not see any pulse, the node Outn is held at VDD and in case a pulse appears at the input D, the node Outn is discharged to Gnd. At the negative edge of the clock, the node Outp is latched.

The pre-charge clock is delayed to avoid hold violation (Figure 128). The proposed pulse detector circuit can detect a pulse of minimum width 10ps.

7.6.2 Pulse Generator

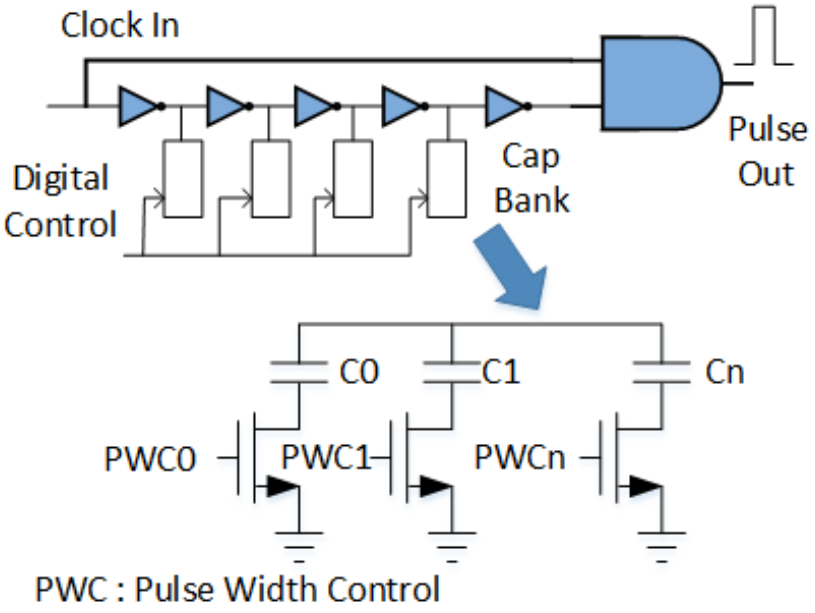


Figure 129: Pulse generator

The pulse generation circuit (Figure 129) is shared among multiple flip-flops across each pipeline stage (Figure 134). The delay values of the inverters in the pulse generation circuit are digitally controlled by using a capacitor bank to precisely vary the pulse width of the generated pulse. In Figure 130 simulation result is shown where a single pulse generator, generating a pulse of 25ps is driving 16 scan flip flops. It requires proper chain of sized inverter (logical effort based inverter chain sizing for driving large load) for pulse routing[158].

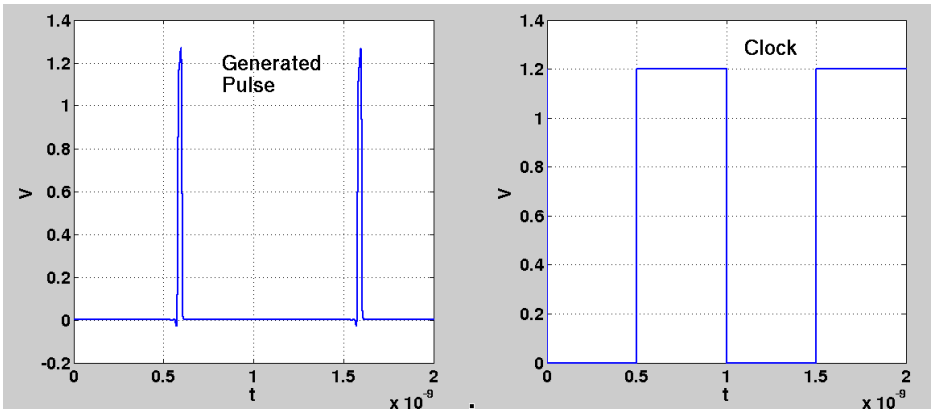


Figure 130: Generated pulse (25ps) from pulse generator loaded with 16 scan flip-flops

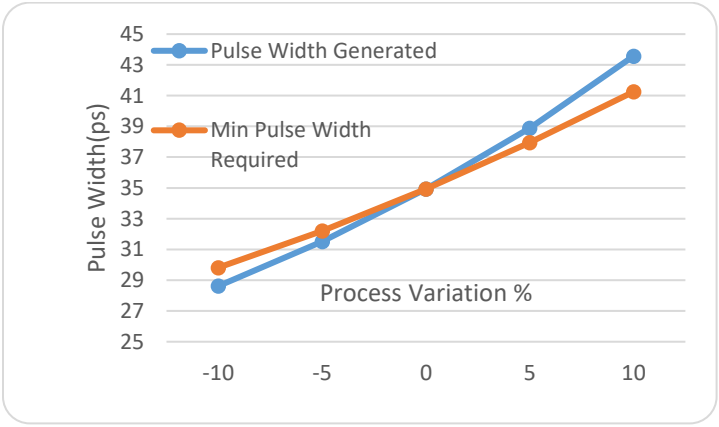


Figure 131: A comparison of pulse width generated and minimum pulse width required

The proposed pulse propagation driven Trojan detection scheme is resilient to systematic process variation to a great extent. An experiment is performed where a pulse generator is driving a pulse through a logic chain (Figure 120). The pulse generator's digital control is so chosen that it generates the minimum required pulse to propagate through the logic chain. The digital control is kept unaltered and the experiment is repeated for various process variation conditions. It is apparent from Figure 131 that the pulse generator almost tracks the minimum required pulse width at various process conditions. Systematic process variation affecting the logic gates and pulse generator circuit gates similarly. If due to

process variation gates are faster (slower) it requires narrower (wider) pulse and as the delay inverters in pulse generator are also faster (slower) the generated pulse is narrower (wider).

7.6.3 Supply Current Sensor for Pulse Detection

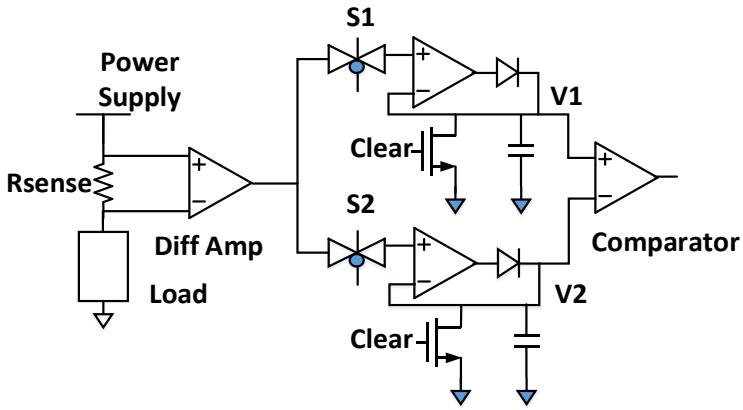


Figure 132: Pulse propagation current sensor

When a pulse propagates through a logic gate, it sinks current from power supply. Weaker the pulse is, lesser is the current drawn from the supply. The peak current difference when a pulse is propagating versus it is not propagating is of the orders of 100 to 1000. Figure 132 is showing the current sensor used to detect the peak current difference. R_{sense} converts the supply current for observables gates to a corresponding voltage. This voltage is amplified by the Differential amplifier. Value of R_{sense} is so chosen that even at maximum current drawing condition voltage droop is not significant (voltage droop is 0.001 volt at maximum current drawing condition in this design).Two peak voltage detectors are used to detect peak voltages, one at +ve cycle of the clock and the other at –ve cycle of the clock. Pulse input is applied at –ve cycle of the clock. So V1 is the peak voltage due to quiescent current of the observable gates and V2 is the peak voltage due to

pulse propagation through any one of the observable gates. Switches S1 & S2 become transparent in +ve and –ve cycle of the clock respectively. One simulation with 20ps current pulse (on amplitude 1 µA, off amplitude 10nA) is shown in Figure 133 to illustrate the operation of the peak current sensor. Clear pulse is used to drain the charges stored in peak detector capacitor to make it ready for next clock cycle test. The comparator goes to logic high when V1 goes above V2 by 50mv. This 50mv offset is kept as a guard band for process variation.

7.7 Application to Digital FSM

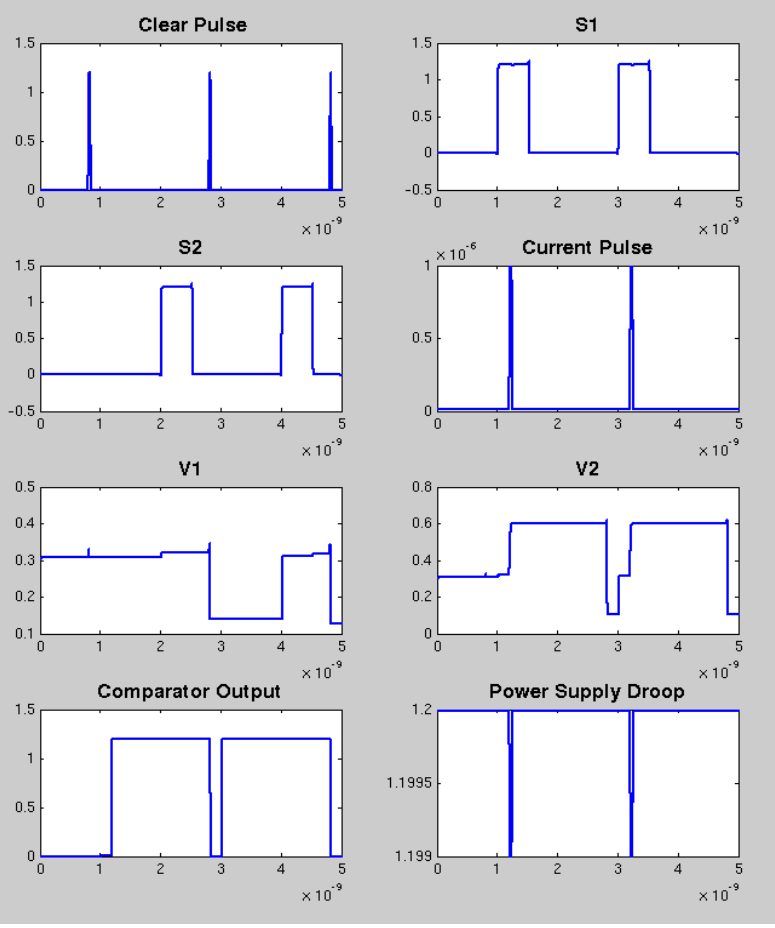


Figure 133: Current sensor simulation result

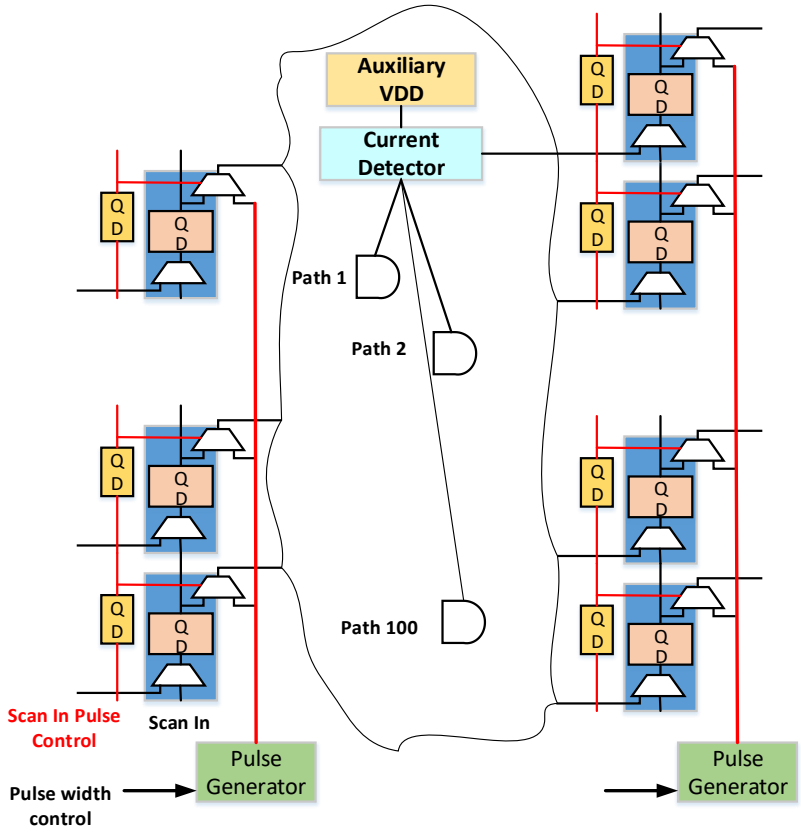


Figure 134: Integrating pulse propagation current detector with scan chain

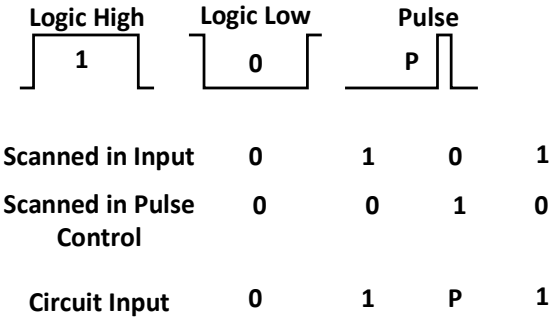


Figure 135: Scanned in values and circuit input

The proposed pulse propagation (both current sensing and voltage sensing) based Trojan detection scheme can easily be incorporated into traditional scan chains available in the digital system for scan testing. A pictorial representation of such a system for pipeline scanning is shown in Figure 134. Similar integration is possible for boundary scanning systems also. In voltage sensing scheme, presence/absence of the pulse is detected at the

input of the scan flop (Figure 127). While in current sensing scheme, one extra scan flop is used in every pipeline to detect existence of Trojan in the previous pipeline logic stage (see Figure 134). For multiple pulse sensing based detection either the DC voltage is taken out (see Figure 136b) or an analog to digital converter is used to integrate it into the scan chain (see Figure 136a). A single pulse generator is shared among all the scan flops of a pipeline stage. It can be shared among multiple pipelines provided different types of pulses are not used simultaneously by two different pipeline stages.

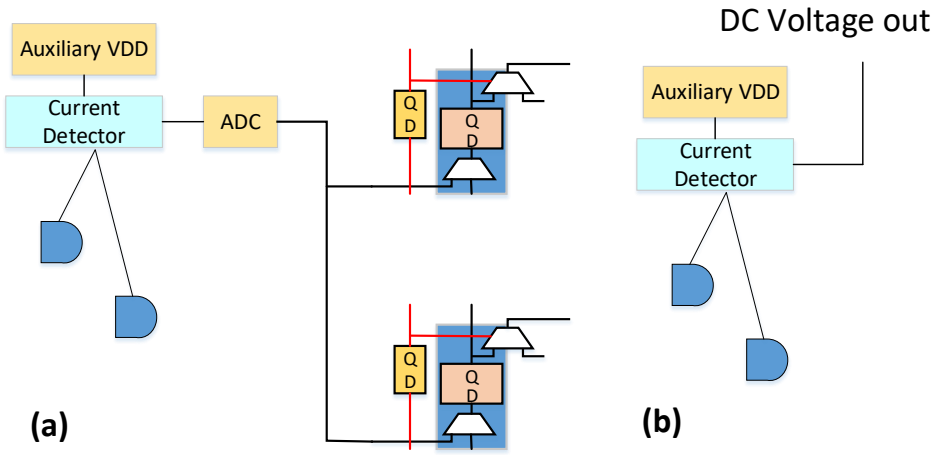


Figure 136: Current sensor integration into the design (a) With an ADC (b) Without ADC, taking dc voltage out

As shown in Figure 134, input to a logic circuit can be a scanned in value or a pulse form the pulse generator depending on scanned in pulse control value. A pulse is applied to the circuit if scanned in pulse control value is logic 1 (See Figure 135 for explanation). It is to be noted that a parallel scan chain is used to scan in pulse control values. In reality these extra flip-flops may not be required. Shadow flip-flop in a scan flip-flop can be used to scan in these pulse control values.

7.8 Possible Attacks on Design For Trojan Circuits:

Pulse generators, pulse detectors will be provided to the foundry as analog IPs. Though it is difficult but not impossible to tamper these analog IPs. We have shown in Figure 134 that probable Trojan infested paths are monitored. We can add some extra paths (these added extra paths may be from original logic circuitry or added chain of inverters) to the monitoring circuit. These paths are sensitized by specific scanned in vectors and pulse propagation (or killing) is controlled by pulse width control vector. As these vector values are not known to the attacker, tampering in pulse generator or current monitoring circuit can be validated during testing. In [147], the authors have proposed to manufacture DFT circuits in a different foundry and using 3D integration, integrate them with the original die. The same approach is applicable here also.

7.9 Experimental Results

In this section we show the efficacy of the proposed pulse based Trojan detection through simulation. In this work 45nm free PDK [150]is used for all circuit level simulations and layout synthesis.

7.9.1 Pulse Detection Using Output Voltage Sensing

The test circuits used to establish the claim are *i) a chain of NAND gates, ii) a ripple carry adder and iii) a 4x4 multiplier*. Process instances are created by altering the threshold voltage of each transistors in the netlist. Here best case (worst case) process variation stands for Vt reduction (increment) by 20% (10% systematic variation & 10 % random variation). A minimally wider pulse propagating through a path is determined for worst case process variation. This pulse is propagated through the same path with best case process variation

gates with a capacitive load at the Trojan affected node. This experiment is repeated with increased value of Trojan capacitance until the pulse gets killed. This is the minimum Trojan capacitance the pulse test can detect in presence of process variation. For delay test the value of the capacitance is obtained which is able to delay the signal as much as the worst case process variation gates were delaying. This is the minimum Trojan capacitance, delay test can detect under process variation.

Table 37: Comparison of proposed pulse propagation test and delay test detection accuracy for nand chain

Logic Depth	Pulse Test	Delay Test	
	Min Cap Detected	Min Cap Detected	Improvement of Pulse Test over Delay Test
5	880aF	1fF	1.13X
10	880aF	2fF	2.27X
16	880aF	3fF	3.40X
22	880aF	4.5fF	5.11X
38	880aF	9fF	10.22X
70	880aF	15f	17.04X

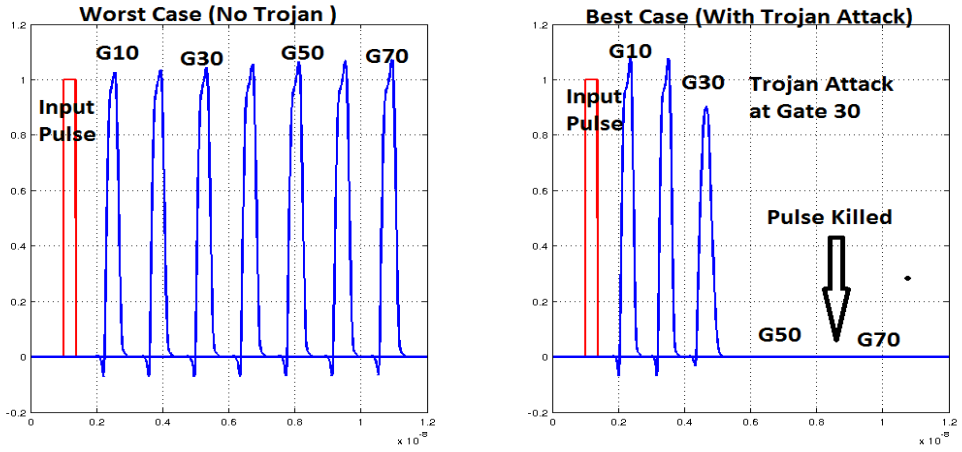


Figure 137: Pulse propagation through nand chain

Results shown in Table 37 is avouching the claim that detection capability in independent of number of gates in a path and also the improvement over delay test is growing as the number of gates in a path is increasing . The similar trend is also observed for ripple carry adder. As the number of FA blocks are increasing, improvement over delay test is increasing proportionally. Simulation results showing minimum detectable capacitance values for delay test and pulse test are shown in Table 37 (NAND chain), Table 38 (Adder) and Table 39 (multiplier).

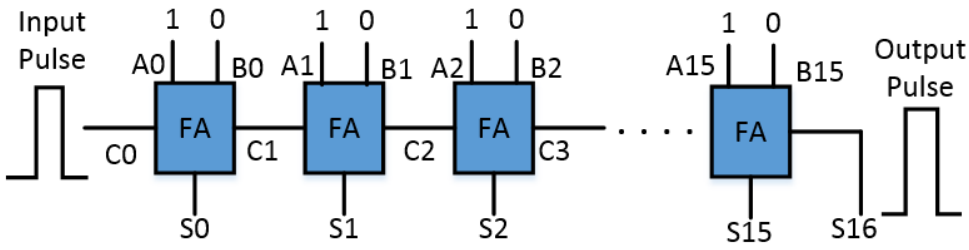


Figure 138. Ripple carry adder

Table 38: Minimum detectable capacitance comparison between proposed pulse propagation and delay method (PP: Pulse propagation method;**DM: Delay method)**

	Minimum Detectable Trojan Capacitance (fF)					
Trojan Cite	C1		C2		C3	
Detection Mechanism	PP	DM	PP	DM	PP	DM
4 bit Adder	1.3	5.5	1.9	5.5	2.5	5.5
8 bit Adder	1.0	13	1.3	13	1.7	13
16 bit Adder	0.95	26	1.2	27	1.5	27

Table 39: Trojan detection in a 4x4 multiplier (D: pulse input)**

			Minimum Detectable Trojan Capacitance (fF)	
Trojan Cite	Input Vector	Pulse Width	Pulse Test	Delay Test
Cout13	A=0D11 B=1111	70ps	0.5	1.3
Sout7	A=111D B=1111	80ps	2.3	4.0
Sout8	A=1011 B=D010	100ps	2.9	7

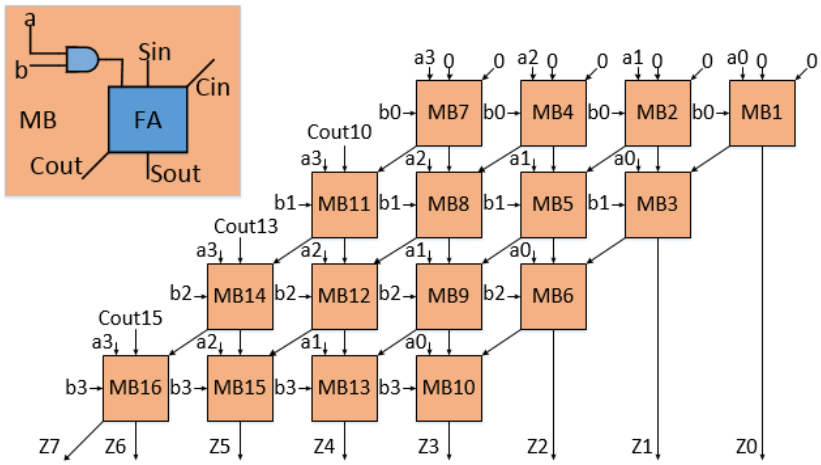


Figure 139: 4X4 Multiplier

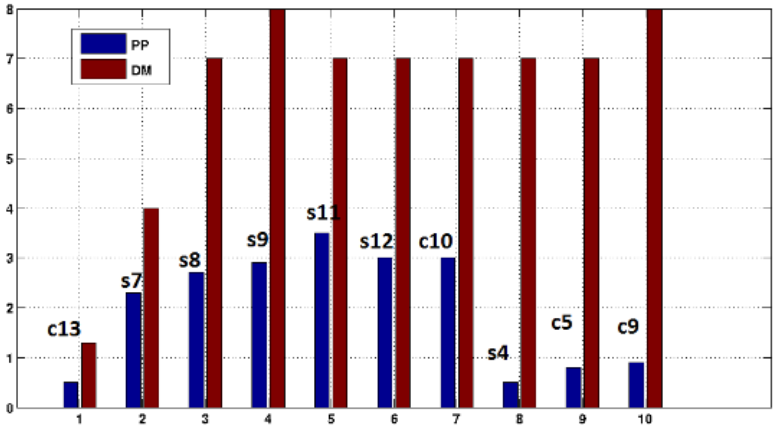


Figure 140: Comparison of minimum capacitance detected by pulse test and delay measurement

7.9.2 Pulse Detection Using Supply Current Sensing

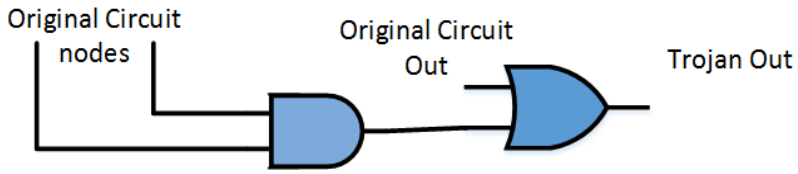


Figure 141: Example Trojan

Trojan activation is a rare event as Trojans inputs are stealthily tapped from very low activation logic nodes. Nature and location of Trojans are not known beforehand. Trojan detection by really activating it, is always an expensive proposition. In this approach, k low activity circuit nodes are identified as probable Trojan nodes and transition probability of those k nodes are increased artificially to reduce Trojan activation time. Let's assume after increasing transition probability the respective transition probabilities are p_{0i} *and* p_{1i}. Then Trojan activation time would be $\prod_{i=1}^{k}(1/\min(p_{0i}, p_{1i})$ cycles. The scheme proposed in this work can detect very small Trojans (one such example is shown in Figure 141) in at

Table 40: Comparison of scan cycles required to detect a Trojan (* ThPr : Threshold transitional probability)

Bench Mark Circuits (ISCAS 85 and TrustHub)	Thpr	# nodes below ThPr	Minimum scan cycles required to detect any Trojan from these nodes	
			This Work	[2]
C432	0.05	50	50	2^{50}
C880	0.05	48	48	2^{48}
C1355	0.05	102	102	2^{102}
C2670	0.1	13	13	2^{13}
C3540	0.05	247	247	2^{247}
C5315	0.05	40	40	2^{40}
C7552	0.05	146	146	2^{146}
RS232-T100	0.03	26	26	2^{26}

most k cycles as k number of pulse propagation will be able to diagnose presence/absence of Trojans in k nodes. The improvement in Trojan detection time is exponential over artificially increasing activity factor. Table 40 is showing this example for various ISCAS 85 and TrustHub[11] bench mark circuits.

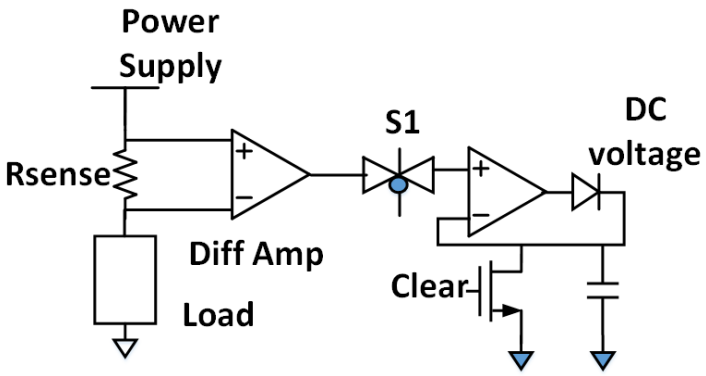


Figure 142: Current sensor for multiple pulse detection based Trojan detection

The current sensor shown in Figure 132 is for sensing current before and during pulse propagation and compare them to detect Trojans. The accuracy of the above said sensor can be improved by sending multiple pulses of varied pulse width and observing the dc voltage for each pulse. The modified current sensor is shown in Figure 142. Four pulses of width 20ps, 30ps 40ps and 50ps were used. Let V20, V30, V40 and V50 be the DC voltage observed at the sensor output respectively. The metric calculated to detect Trojan is shown in equation 6.

$$Trojan\ Metric = \sum_{i=2}^{5} \frac{V_{50}}{V_{i0}} - \frac{V_{20}}{V_{i0}} \tag{120}$$

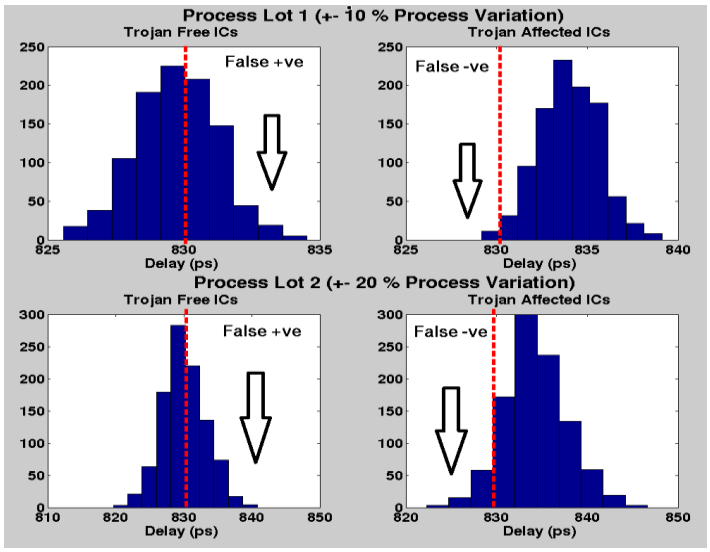


Figure 143: Monte Carlo simulation result (delay measurement)

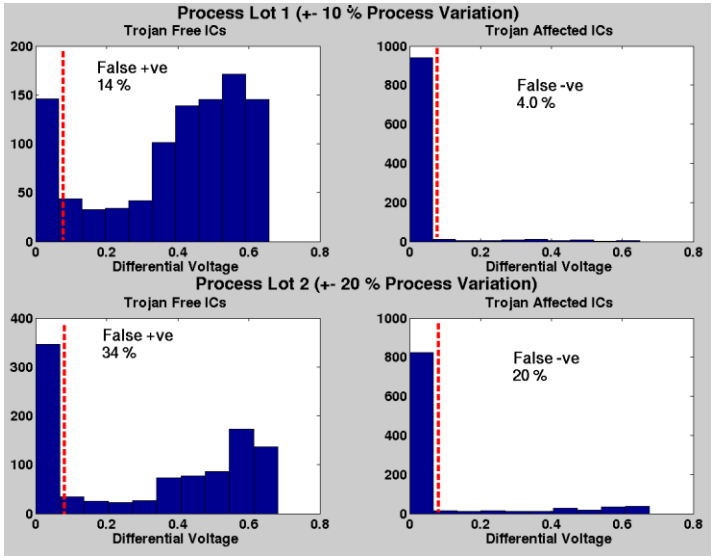


Figure 144: Monte Carlo simulation (single pulse propagation technique)

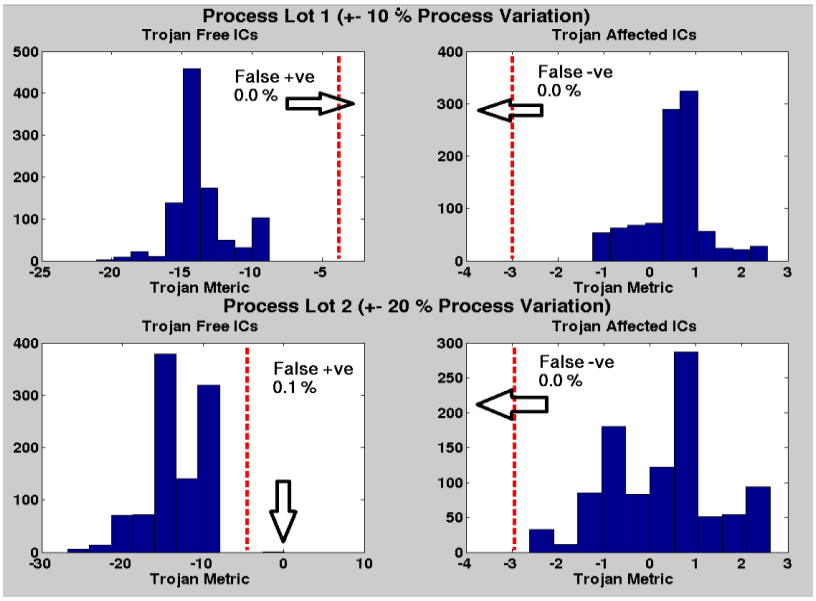


Figure 145: Monte Carlo simulation (multiple pulse propagation technique)

To compare pulse technique with delay based techniques we took a random path from c1355 netlist and created two process lots. For each process lots threshold voltage (vt) of every transistor in the netlist is sampled from a normal distribution where mean is nominal vt and standard deviation is 0.1vt (0.2vt for process lot 2). Each lot contains 2000 process varied instances out of which 1000 were Trojan affected. Monte Carlo simulation results for all those 2000 devices from each process lots are shown in Figure 143, Figure 144 and Figure 145. Errors in prediction for Trojan occurrence (false +ve, false –ve, average prediction error) is tabulated and shown in Table 41. For a fair comparison among the detection techniques we have not employed any process calibration. Process calibration will help all the contending Trojan detection techniques. From Figure 143, we see that delays are difficult to discern in presence of process variation and average error of prediction is high, 22.25% (process lot 1) and 28.9 %(process lot 2). Similarly for single pulse propagation (see Table 41) average error is relatively high 9% (process lot 1) and 22% (process lot 2). For both the techniques prediction accuracy is decreasing as process

Table 41: Miss prediction in Trojan detection for various techniques

	Process lot 1 (+- 10% random vt variation)			Process lot 2 (+- 20% random vt variation)		
	False +ve	False -ve	Avg error	False +ve	False -ve	Avg error
Delay Based Measurement	43.7%	0.8%	22.25 %	48.3%	9.5%	28.9%
Pulse Detection Based Measurement (Single Pulse) [1]	14%	4%	9%	34%	20%	27%
Pulse Detection Based Measurement (Multiple Pulse) [This Work]	0%	0%	0%	0.1%	0%	0.05%

variation is increasing (process lot 1 to process lot 2), which is obvious and self-explanatory. We see dramatically improvement in results in Figure 145 for multiple pulse propagation technique proposed in this work. Trojan metric proposed in equation (120) can easily discern between Trojan affected and Trojan free ICs. Average prediction errors are 0.0% (process lot 1) and 0.05% (process lot 2). The above results establish the efficacy of the proposed Trojan detection scheme by simulation.

We will explain the process of Trojan detection by taking another TrustHub[11] example circuit RS232-T1000. Trojan activation probability here is 3.55e-13. It is almost impossible to activate the Trojan and detect it with functional testing. We ran sufficiently large number of input patterns and find probable low transition nodes. In this example we set threshold probability of 0.03 and implemented pulse based Trojan monitoring for those nodes. We monitored total 26 circuit nodes in this example. Threshold probability determines the number of nodes to be observed and thus directly controls extra area requirement for security. In this example node “iXMIT_N_CTRL_2_” flags Trojan.

7.9.3 Dopant Trojan Detection

As explained in section 3, dopant Trojans short circuit node to either VDD or Gnd. We took the same random path from c1355 netlist (Trojan experiment of section 8.2) and create a short to VDD with 10Ω resistance. As expected the pulse will not propagate and the current sensor will be able to detect presence of Trojans in the circuit without any difficulty (see Figure 146).

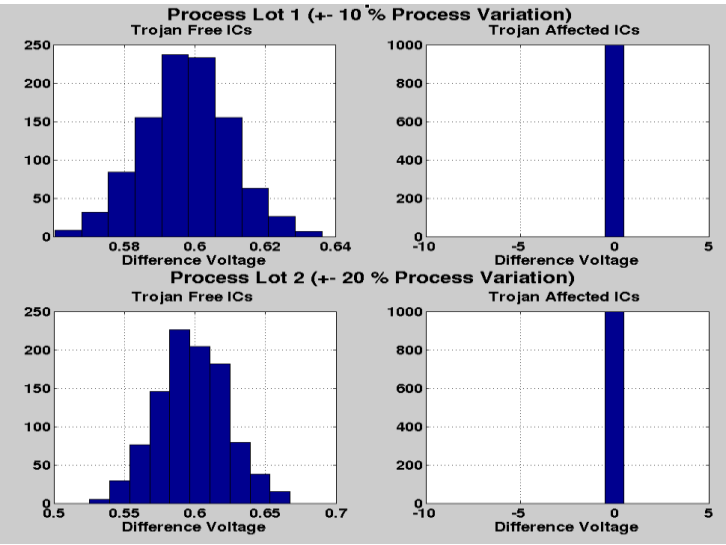


Figure 146: Monte Carlo simulation result (dopant Trojan)

Extra analog circuitry required for the proposed multiple pulse propagation based Trojan detection are laid out and shown in Figure 147 (pulse generator) and in Figure 148 (current sensor). ISCAS bench mark circuits are synthesized using the 45 nm free PDK [150] and an area comparison is shown in Table 42. Power consumption (shown in Table 43) of the extra analog circuitry is of not prime concern as they will only be powered on during testing. Synopsis's design compiler is used for logic synthesis.

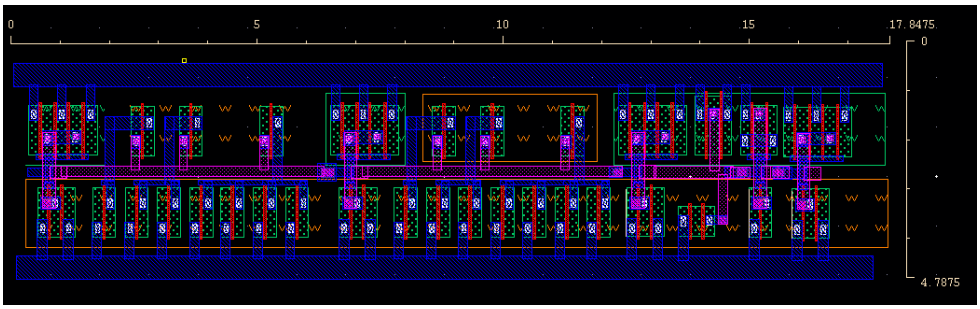


Figure 147: Pulse generator layout

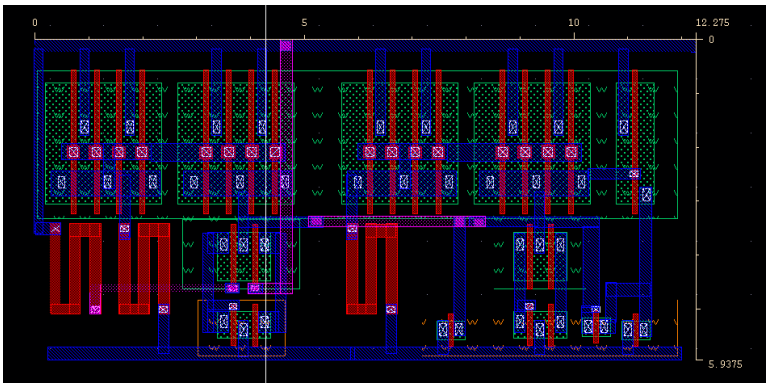


Figure 148: Current sensor layout

Table 42: Area overhead of proposed Trojan detection scheme for bench mark circuits

Bench Mark Circuit (ISCAS 85 and TrustHub)	Synthesized Area (μm^2)	Area Overhead for Proposed Trojan Detection Scheme
C3540	2773.55	5.70%
C5315	3605.15	4.38%
C6288	2914.03	5.42%
C7552	3640.36	4.34%
S35932	28906.06	1.04%

Table 43: Power overhead of proposed Trojan detection scheme for bench mark circuits

Bench Mark Circuit (ISCAS 85 and TrustHub)	Power Estimation from synthesis (μW)	Power Overhead for Proposed Trojan Detection Scheme (μW)	
		During Testing	During normal Operation
C3540	705.83	0.71%	0.21%
C5315	1270.0	0.39%	0.12%
C6288	2367.1	0.21%	0.06%
C7552	1400.0	0.36%	0.11%
S35932	7484.3	0.06%	0.01%

7.10 Conclusion

In this work the authors have given a comprehensive pulse propagation driven Trojan detection scheme built on their previous work of [1, 144]. While most of the Trojan detection techniques lose diagnostic accuracy amidst process variation, the proposed technique is independent of circuit size. No Trojan free manufactured ICs are required for model building and no process variation calibration is needed. The current sensor built in [1] is modified and also the Trojan detection scheme proposed in [1, 144] is modified in order to achieve higher accuracy in detection. Power and area requirements of the additional circuitry used for Trojan detection are given. The entire Trojan detection scheme can be easily integrated with existing JTAG scan chain testing protocol. Advantages of the proposed Trojan detection scheme over the other state of the art Trojan detection schemes (Delay based detection, artificially increasing activity factors of low activity nodes) have been compared and simulation results corroborate the efficacy of the proposed Trojan detection scheme.

REFERENCES

[1] Sabyasachi Deyati , Barry J. Muldrey, Abhijit Chatterjee, "Trojan Detection in Digital Systems Using Current Sensing of Pulse Propagation in Logic Gates " in *ISQED*, Santa Clara USA, 2016.

[2] H. Salmani, M. Tehranipoor, and J. Plusquellic, "New design strategy for improving hardware Trojan detection and reducing Trojan activation time," in *Hardware-Oriented Security and Trust, 2009. HOST '09. IEEE International Workshop on*, 2009, pp. 66-73.

[3] S. Deyati, B. J. Muldrey, and A. Chatterjee, "Adaptive testing of analog/RF circuits using hardware extracted FSM models," in *2016 IEEE 34th VLSI Test Symposium (VTS)*, 2016, pp. 1-6.

[4] S. Deyati, B. J. Muldrey, and A. Chatterjee, "TRAP: Test Generation Driven Classification of Analog/RF ICs Using Adaptive Probabilistic Clustering Algorithm," in *2016 29th International Conference on VLSI Design and 2016 15th International Conference on Embedded Systems (VLSID)*, 2016, pp. 463-468.

[5] A. Chatterjee, S. Deyati, B. Muldrey, S. Devarakond, and A. Banerjee, "Validation signature testing: A methodology for post-silicon validation of analog/mixed-signal circuits," in *2012 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2012, pp. 553-556.

[6] S. Deyati, A. Banerjee, and A. Chatterjee, "Pilot symbol driven monitoring of electrical degradation in RF transmitter systems using model anomaly diagnosis," in *2012 IEEE 18th International On-Line Testing Symposium (IOLTS)*, 2012, pp. 142-145.

[7] S. Deyati, A. Banerjee, B. J. Muldrey, and A. Chatterjee, "VAST: Post-Silicon VAlidation and Diagnosis of RF/Mixed-Signal Circuits Using Signature Tests," in *VLSI Design and 2013 12th International Conference on Embedded Systems (VLSID), 2013 26th International Conference on*, 2013, pp. 314-319.

[8] A. Chatterjee, S. Deyati, and B. J. Muldrey, "Post silicon validation of analog/mixed signal/RF circuits and systems: recent advances," in *2016 IEEE 21st International Mixed-Signal Testing Workshop (IMSTW)*, 2016, pp. 1-6.

[9] S. Deyati, B. J. Muldrey, A. Banerjee, and A. Chatterjee, "Atomic model learning: A machine learning paradigm for post silicon debug of RF/analog circuits," in *2014 IEEE 32nd VLSI Test Symposium (VTS)*, 2014, pp. 1-6.

[10] S. Deyati, B. Muldrey, and A. Chatterjee, "BISCC: Efficient pre through post silicon validation of mixed-signal/RF systems using built in state consistency checking," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, 2017, pp. 274-277.

[11] S. Deyati, B. J. Muldrey, A. D. Singh, and A. Chatterjee, "Challenge Engineering and Design of Analog Push Pull Amplifier Based Physically Unclonable Function for Hardware Security," in *2015 IEEE 24th Asian Test Symposium (ATS)*, 2015, pp. 127-132.

[12] S. Deyati, B. J. Muldrey, A. Singh, and A. Chatterjee, "High Resolution Pulse Propagation Driven Trojan Detection in Digital Logic: Optimization Algorithms and Infrastructure," in *2014 IEEE 23rd Asian Test Symposium*, 2014, pp. 200-205.

[13] S. Deyati, B. J. Muldrey, and A. Chatterjee, "Targeting hardware trojans in mixed-signal circuits for security," in *2016 IEEE 21st International Mixed-Signal Testing Workshop (IMSTW)*, 2016, pp. 1-4.

[14] S. Deyati, B. J. Muldrey, and A. Chatterjee, "Trojan detection in digital systems using current sensing of pulse propagation in logic gates," in *2016 17th International Symposium on Quality Electronic Design (ISQED)*, 2016, pp. 350-355.

[15] R. Voorakaranam, R. Newby, S. Cherubal, B. Cometta, T. Kuehl, D. Majernik, *et al.*, "Production deployment of a fast transient testing methodology for analog circuits: case study and results," in *Test Conference, 2003. Proceedings. ITC 2003. International*, 2003, pp. 1174-1181.

[16] R. Voorakaranam, S. S. Akbay, S. Bhattacharya, S. Cherubal, and A. Chatterjee, "Signature Testing of Analog and RF Circuits: Algorithms and Methodology," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 54, pp. 1018-1031, 2007.

[17] P. N. Variyam, S. Cherubal, and A. Chatterjee, "Prediction of analog performance parameters using fast transient testing," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 21, pp. 349-361, 2002.

[18] V. Natarajan, C. Hyun Woo, A. Banerjee, S. Sen, A. Chatterjee, G. Srinivasan, *et al.*, "Low Cost EVM Testing of Wireless RF SoC Front-Ends Using Multitones," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 31, pp. 1088-1101, 2012.

[19] E. Acar and S. Ozev, "Go/No-Go Testing of VCO Modulation RF Transceivers Through the Delayed-RF Setup," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 15, pp. 37-47, 2007.

[20] E. S. Erdogan and S. Ozev, "Detailed Characterization of Transceiver Parameters Through Loop-Back-Based BiST," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 18, pp. 901-911, 2010.

[21] S. Bhattacharya and A. Chatterjee, "Optimized wafer-probe and assembled package test design for analog circuits," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 10, pp. 303-329, 2005.

[22] A. V. Gomes, "Alternate Test Generation for Detection of Parametric Faults," PhD, ECE, Georgia Institute of Technology, 2003.

[23] C. Hsiu-Ming, C. Kwang-Ting, Z. Wangyang, L. Xin, and K. M. Butler, "Test cost reduction through performance prediction using virtual probe," in *Test Conference (ITC), 2011 IEEE International*, 2011, pp. 1-9.

[24] P. N. Variyam and A. Chatterjee, "Specification-driven test generation for analog circuits," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 19, pp. 1189-1201, 2000.

[25] S. Bhattacharya, A. Halder, G. Srinivasan, and A. Chatterjee, "Alternate Testing of RF Transceivers Using Optimized Test Stimulus for Accurate Prediction of System Specifications," *Journal of Electronic Testing*, vol. 21, pp. 323-339, 2005/06/01 2005.

[26] G. Srinivasan, S. Bhattacharya, S. Cherubal, and A. Chatterjee, "Fast specification test of TDMA power amplifiers using transient current measurements," *Computers and Digital Techniques, IEE Proceedings* -, vol. 152, pp. 632-642, 2005.

[27] S. Benner and O. Boroffice, "Optimal production test times through adaptive test programming," in *Test Conference, 2001. Proceedings. International*, 2001, pp. 908-915.

[28] P. Maxwell, "Adaptive Testing: Dealing with Process Variability," *IEEE Design & Test of Computers,* vol. 28, pp. 41-49, 2011.

[29] E. Yilmaz, S. Ozev, and K. M. Butler, "Per-Device Adaptive Test for Analog/RF Circuits Using Entropy-Based Process Monitoring," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* vol. 21, pp. 1116-1128, 2013.

[30] H. G. Stratigopoulos and S. Mir, "Adaptive Alternate Analog Test," *IEEE Design & Test of Computers,* vol. 29, pp. 71-79, 2012.

[31] A. Genz, Bretz, Frank, *Computation of Multivariate Normal and t Probabilities*: Springer, 2009.

[32] M. Bensimhoun, "N-DIMENSIONAL CUMULATIVE FUNCTION, AND OTHER USEFUL FACTS ABOUT GAUSSIANS AND NORMAL DENSITIES " 2009.

[33] R. T. Trevor Hastie , Jerome Friedman, *The Elements of Statistical Learning:Data Mining, Inference, and Prediction.*: Springer, February 2009.

[34] G. J. Szekely, M. L. Rizzo, and N. K. Bakirov, "Measuring and testing dependence by correlation of distances," pp. 2769-2794, 2007/12 2007.

[35] E. Parzen, "On Estimation of a Probability Density Function and Mode," *The Annals of Mathematical Statistics,* vol. 33, 1962.

[36] T. Cacoullos, "Estimation of a multivariate density," *Annals of the Institute of Statistical Mathematics,* vol. 18, pp. 179-189, 1966/12/01 1966.

[37] V. Cheung and K. Cannons. An Introduction to Probabilistic Neural Networks [Online]. Available: www.vincentcheung.ca/research/research/presentations/PNN.pdf

[38] E. Yilmaz, S. Ozev, and K. M. Butler, "Adaptive multidimensional outlier analysis for analog and mixed signal circuits," in *Test Conference (ITC), 2011 IEEE International*, 2011, pp. 1-8.

[39] H. G. Stratigopoulos, S. Mir, E. Acar, and S. Ozev, "Defect Filter for Alternate RF Test," in *Test Symposium, 2009 14th IEEE European*, 2009, pp. 101-106.

[40] A. Haldar, "Efficient Alternate Test Generation for RF Transceiver Architectures," 2006.

[41] A. Halder and A. Chatterjee, "Low-cost production testing of wireless transmitters," in *VLSI Design, 2006. Held jointly with 5th International Conference on Embedded Systems and Design., 19th International Conference on*, 2006, p. 6 pp.

[42] A. V. Karthik and J. Roychowdhury, "ABCD-L: Approximating continuous linear systems using Boolean models," in *Design Automation Conference (DAC), 2013 50th ACM / EDAC / IEEE*, 2013, pp. 1-9.

[43] A. V. Karthik, S. Ray, P. Nuzzo, A. Mishchenko, R. Brayton, and J. Roychowdhury, "ABCD-NL: Approximating Continuous non-linear dynamical systems using purely Boolean models for analog/mixed-signal verification," in

Design Automation Conference (ASP-DAC), 2014 19th Asia and South Pacific, 2014, pp. 250-255.
[44] [Online]. Available: http://pdfserv.maximintegrated.com/en/an/AN1838.pdf
[45] *Predictive Technology Model*. Available: http://ptm.asu.edu/
[46] L. Ting, H. Hu, and S. Yihe, "ISTA: An embedded architecture for post-silicon validation in processors," in *2009 IEEE 8th International Conference on ASIC*, 2009, pp. 593-596.
[47] J. Keshava, N. Hakim, and C. Prudvi, "Post-silicon validation challenges: How EDA and academia can help," in *Design Automation Conference*, 2010, pp. 3-7.
[48] S. Mitra, S. A. Seshia, and N. Nicolici, "Post-silicon validation opportunities, challenges and recent advances," in *Design Automation Conference*, 2010, pp. 12-17.
[49] A. DeOrio, I. Wagner, and V. Bertacco, "Dacota: Post-silicon validation of the memory subsystem in multi-core designs," in *2009 IEEE 15th International Symposium on High Performance Computer Architecture*, 2009, pp. 405-416.
[50] S. Shedabale, H. Ramakrishnan, G. Russell, A. Yakovlev, and S. Chattopadhyay, "Statistical modelling of the variation in advanced process technologies using a multi-level partitioned response," *IET Circuits, Devices & Systems*, vol. 2, pp. 451-464, 2008.
[51] S. V. Kumar, "Reliability aware and variation aware CAD Techniques," PhD, Electrical Engineering, UNIVERSITY OF MINNESOTA, 2009.
[52] H. Chang and K. Kundert, "Verification of Complex Analog and RF IC Designs," *Proceedings of the IEEE*, vol. 95, pp. 622-639, 2007.
[53] K. Muhammad, T. Murphy, and R. B. Staszewski, "Verification of RF SoCs: RF, analog, baseband and software," in *IEEE Radio Frequency Integrated Circuits (RFIC) Symposium, 2006*, 2006, pp. 4 pp.-364.
[54] K. Kundert, H. Chang, D. Jefferies, G. Lamant, E. Malavasi, and F. Sendig, "Design of mixed-signal systems-on-a-chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, pp. 1561-1571, 2000.
[55] E. Chou and B. Sheu, "Nanometer mixed-signal system-on-a-chip design," *IEEE Circuits and Devices Magazine*, vol. 18, pp. 7-17, 2002.
[56] N. Chandra and G. W. Roberts, "Top-down analog design methodology using Matlab and Simulink," in *ISCAS 2001. The 2001 IEEE International Symposium on Circuits and Systems (Cat. No.01CH37196)*, 2001, pp. 319-322 vol. 5.
[57] O. V. International, *Verilog-AMS Language Reference Manual: Analog & Mixed-signal Extension to Verilog HDL ; Version 2.0*: Open Verilog International, 2000.
[58] E. Christen and K. Bakalar, "VHDL-AMS-a hardware description language for analog and mixed-signal applications," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, pp. 1263-1272, 1999.
[59] C. J. Myers, R. R. Harrison, D. Walter, N. Seegmiller, and S. Little, "The Case for Analog Circuit Verification," *Electronic Notes in Theoretical Computer Science*, vol. 153, pp. 53-63, 2006/06/20 2006.
[60] M. Freibothe, J. Schönherr, and B. Straube, "Formal Verification of the Quasi-Static Behavior of Mixed-Signal Circuits by Property Checking," *Electronic Notes in Theoretical Computer Science*, vol. 153, pp. 23-35, 2006/06/20 2006.

[61] R. Jeongjin, S. Seshadri, and J. A. Abraham, "Verification of Delta-Sigma converters using adaptive regression modeling," in *IEEE/ACM International Conference on Computer Aided Design. ICCAD - 2000. IEEE/ACM Digest of Technical Papers (Cat. No.00CH37140)*, 2000, pp. 182-187.

[62] A. Balivada, Y. Hoskote, and J. A. Abraham, "Verification of transient response of linear analog circuits," in *Proceedings 13th IEEE VLSI Test Symposium*, 1995, pp. 42-47.

[63] T. H. Lee, *The Design of CMOS Radio-Frequency Integrated Circuits*, 2 ed. Cambridge: Cambridge University Press, 2003.

[64] B. Razavi, *RF Microelectronics (2nd Edition) (Prentice Hall Communications Engineering and Emerging Technologies Series)*: Prentice Hall Press, 2011.

[65] A. Banerjee, S. Sen, S. K. Devarakond, and A. Chatterjee, "Automatic test stimulus generation for accurate diagnosis of RF systems using transient response signatures," in *29th VLSI Test Symposium*, 2011, pp. 58-63.

[66] A. Banerjee, V. Natarajan, S. Sen, A. Chatterjee, G. Srinivasan, and S. Bhattacharya, "Optimized Multitone Test Stimulus Driven Diagnosis of RF Transceivers Using Model Parameter Estimation," in *2011 24th Internatioal Conference on VLSI Design*, 2011, pp. 274-279.

[67] A. Nahir, A. Ziv, M. Abramovici, A. Camilleri, R. Galivanche, B. Bentley, *et al.*, "Bridging pre-silicon verification and post-silicon validation," in *Design Automation Conference (DAC), 2010 47th ACM/IEEE*, 2010, pp. 94-95.

[68] S. Ray and W. A. Hunt, "Connecting pre-silicon and post-silicon verification," in *Formal Methods in Computer-Aided Design, 2009. FMCAD 2009*, 2009, pp. 160-163.

[69] M. H. Neishaburi and Z. Zilic, "A distributed AXI-based platform for post-silicon validation," in *VLSI Test Symposium (VTS), 2011 IEEE 29th*, 2011, pp. 8-13.

[70] D. Lin, T. Hong, Y. Li, F. Fallah, D. S. Gardner, N. Hakim, *et al.*, "Overcoming post-silicon validation challenges through Quick Error Detection (QED)," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2013*, 2013, pp. 320-325.

[71] A. Chatterjee, S. Deyati, B. Muldrey, S. Devarakond, and A. Banerjee, "Validation signature testing: A methodology for post-silicon validation of analog/mixed-signal circuits," in *Computer-Aided Design (ICCAD), 2012 IEEE/ACM International Conference on*, 2012, pp. 553-556.

[72] B. Muldrey, S. Deyati, M. Giardino, and A. Chatterjee, "RAVAGE: Post-silicon validation of mixed signal systems using genetic stimulus evolution and model tuning," in *VLSI Test Symposium (VTS), 2013 IEEE 31st*, 2013, pp. 1-6.

[73] N. Z. Hakim, A. Bhaduri, K. Donepudi, and S. Bodapati, "A hybrid electrical-behavioral modeling approach for pre- and post-silicon electrical validation," in *Custom Integrated Circuits Conference (CICC), 2012 IEEE*, 2012, pp. 1-5.

[74] G. Chenjie, E. Chiprout, and L. Xin, "Efficient moment estimation with extremely small sample size via bayesian inference for analog/mixed-signal validation," in *Design Automation Conference (DAC), 2013 50th ACM / EDAC / IEEE*, 2013, pp. 1-7.

[75] J. Y. Goulermas, P. Liatsis, Z. Xiao-Jun, and P. Cook, "Density-Driven Generalized Regression Neural Networks (DD-GRNN) for Function Approximation," *Neural Networks, IEEE Transactions on*, vol. 18, pp. 1683-1696, 2007.
[76] Z. Dongling, T. Yingjie, and Z. Peng, "Kernel-Based Nonparametric Regression Method," in *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT '08. IEEE/WIC/ACM International Conference on*, 2008, pp. 410-413.
[77] C. M. Bishop, *Pattern Recognition and Machine Learning*: Springer, 2006.
[78] J. Jae Woong, S. Ozev, S. Sen, and T. M. Mak, "Measurement of envelope/phase path delay skew and envelope path bandwidth in polar transmitters," in *VLSI Test Symposium (VTS), 2013 IEEE 31st*, 2013, pp. 1-6.
[79] J. C. Pedro, J. A. Garcia, and P. M. Cabral, "Nonlinear Distortion Analysis of Polar Transmitters," *Microwave Theory and Techniques, IEEE Transactions on*, vol. 55, pp. 2757-2765, 2007.
[80] K. Tae-Woo, L. Min-Chul, and C. Gyu-Hyeong, "A 2 W CMOS Hybrid Switching Amplitude Modulator for EDGE Polar Transmitters," *Solid-State Circuits, IEEE Journal of*, vol. 42, pp. 2666-2676, 2007.
[81] C. Yong-Chang, S.-S. Yoo, and Y. Hyung-Joun, "A fully digital polar transmitter using a digital-to-time converter for high data rate system," in *Radio-Frequency Integration Technology, 2009. RFIT 2009. IEEE International Symposium on*, 2009, pp. 56-59.
[82] A. V. Agrawal and R. Mehra, "Reconfigurable Design of Rectangular to Polar Converter using Linear Convergence. . Published by Foundation of Computer Science, New York, USA," *International Journal of Computer Applications* vol. 50, pp. 23-27, July 2012.
[83] Available: http://www.mathworks.com/help/pdf_doc/nnet/nnet_ug.pdf
[84] A. Salem, "Semi-formal verification of VHDL-AMS descriptions," in *Circuits and Systems, 2002. ISCAS 2002. IEEE International Symposium on*, 2002, pp. V-333-V-336 vol.5.
[85] S. Gupta, B. H. Krogh, and R. A. Rutenbar, "Towards formal verification of analog designs," in *Computer Aided Design, 2004. ICCAD-2004. IEEE/ACM International Conference on*, 2004, pp. 210-217.
[86] B. Suparjo, A. Ley, A. Cron, and H. Ehrenberg, "Analog Boundary-Scan Description Language (ABSDL) for Mixed-Signal Board Test," in *Test Conference, 2006. ITC '06. IEEE International*, 2006, pp. 1-9.
[87] "IEEE Standard for a Mixed-Signal Test Bus," *IEEE Std 1149.4-2010 (Revision of IEEE Std 1149.4-1999)*, pp. 1-116, 2011.
[88] M. Soma, "Structure and concepts for current-based analog scan," in *Custom Integrated Circuits Conference, 1995., Proceedings of the IEEE 1995*, 1995, pp. 517-520.
[89] M. Soma, T. M. Bocek, T. D. Vu, and J. D. Moffatt, "Experimental results for current-based analog scan," in *Test Conference, 1997. Proceedings., International*, 1997, pp. 768-775.
[90] R. Vasudevamurthy, P. K. Das, and B. Amrutur, "A mostly-digital analog scan-out chain for low bandwidth voltage measurement for analog IP test," in *Circuits and Systems (ISCAS), 2011 IEEE International Symposium on*, 2011, pp. 2035-2038.

[91] A. Zjajo, H. J. Bergveld, R. Schuttert, and J. P. d. Gyvez, "Power-scan chain: design for analog testability," in *Test Conference, 2005. Proceedings. ITC 2005. IEEE International*, 2005, pp. 8 pp.-83.

[92] S. Deyati, B. J. Muldrey, A. Banerjee, and A. Chatterjee, "Atomic model learning: A machine learning paradigm for post silicon debug of RF/analog circuits," in *VLSI Test Symposium (VTS), 2014 IEEE 32nd*, 2014, pp. 1-6.

[93] B. Querbach, R. Khanna, D. Blankenbeckler, Y. Zhang, R. T. Anderson, D. G. Ellis, *et al.*, "A reusable BIST with software assisted repair technology for improved memory and IO debug, validation and test time," in *Test Conference (ITC), 2014 IEEE International*, 2014, pp. 1-10.

[94] X. Shi and N. Nicolici, "On-chip generation of uniformly distributed constrained-random stimuli for post-silicon validation," in *Computer-Aided Design (ICCAD), 2015 IEEE/ACM International Conference on*, 2015, pp. 808-815.

[95] N. Oh, P. P. Shirvani, and E. J. McCluskey, "Error detection by duplicated instructions in super-scalar processors," *IEEE Transactions on Reliability*, vol. 51, pp. 63-75, 2002.

[96] D. Lin, T. Hong, Y. Li, S. E, S. Kumar, F. Fallah, *et al.*, "Effective Post-Silicon Validation of System-on-Chips Using Quick Error Detection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 33, pp. 1573-1590, 2014.

[97] A. Garg and T. T. Kim, "Design of SRAM PUF with improved uniformity and reliability utilizing device aging effect," in *Circuits and Systems (ISCAS), 2014 IEEE International Symposium on*, 2014, pp. 1941-1944.

[98] S. Sun, T. S. Rappaport, R. W. Heath, A. Nix, and S. Rangan, "Mimo for millimeter-wave wireless communications: beamforming, spatial multiplexing, or both?," *IEEE Communications Magazine,* vol. 52, pp. 110-121, 2014.

[99] S. Kutty and D. Sen, "Beamforming for Millimeter Wave Communications: An Inclusive Survey," *IEEE Communications Surveys & Tutorials,* vol. 18, pp. 949-973, 2016.

[100] K.-P. Ho, S. Cheng, and J. Liu, "MIMO Beamforming in Millimeter Wave Directional Wi-Fi," 2014.

[101] M. Elkourdi, B. Peköz, E. Güvenkaya, and H. Arslan, "Waveform design principles for 5G and beyond," in *2016 IEEE 17th Annual Wireless and Microwave Technology Conference (WAMICON)*, 2016, pp. 1-6.

[102] M. Agiwal, A. Roy, and N. Saxena, "Next Generation 5G Wireless Networks: A Comprehensive Survey," *IEEE Communications Surveys & Tutorials,* vol. 18, pp. 1617-1655, 2016.

[103] V. Natarajan, R. Senguttuvan, S. Sen, and A. Chatterjee, "Built-in Test Enabled Diagnosis and Tuning of RF Transmitter Systems," *VLSI Design*, vol. 2008, p. 10, 2008.

[104] D. Han, B. S. Kim, and A. Chatterjee, "DSP-Driven Self-Tuning of RF Circuits for Process-Induced Performance Variability," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* vol. 18, pp. 305-314, 2010.

[105] V. Natarajan, S. Sen, A. Banerjee, A. Chatterjee, G. Srinivasan, and F. Taenzler, "Analog Signature- Driven Postmanufacture Multidimensional Tuning of RF Systems," *IEEE Des. Test,* vol. 27, pp. 6-17, 2010.

[106] D. Maliuk and Y. Makris, "An Experimentation Platform for On-Chip Integration of Analog Neural Networks: A Pathway to Trusted and Robust Analog/RF ICs," *IEEE Transactions on Neural Networks and Learning Systems,* vol. 26, pp. 1721-1734, 2015.
[107] J. Wagner, R. Wolf, N. Joram, and F. Ellinger, "0.5-9 GHz CMOS variable gain amplifier with control linearization, low phase variations and self-matching," in *2014 9th European Microwave Integrated Circuit Conference*, 2014, pp. 100-103.
[108] E. Acar and S. Ozev, "Low Cost MIMO Testing for RF Integrated Circuits," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* vol. 18, pp. 1348-1356, 2010.
[109] S. Devarakond, D. Banerjee, A. Banerjee, S. Sen, and A. Chatterjee, "Efficient system-level testing and adaptive tuning of MIMO-OFDM wireless transmitters," in *2013 18th IEEE European Test Symposium (ETS)*, 2013, pp. 1-6.
[110] V. Natarajan, H. W. Choi, A. Banerjee, S. Sen, A. Chatterjee, G. Srinivasan, *et al.*, "Low Cost EVM Testing of Wireless RF SoC Front-Ends Using Multitones," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 31, pp. 1088-1101, 2012.
[111] A. Nassery, S. Ozev, and M. Slamani, "Analytical modeling for EVM in OFDM transmitters including the effects of IIP3, I/Q imbalance, noise, AM/AM and AM/PM distortion," in *2013 18th IEEE European Test Symposium (ETS)*, 2013, pp. 1-6.
[112] R. H. Byrd, J. C. Gilbert, and J. Nocedal, "A trust region method based on interior point techniques for nonlinear programming," *Mathematical Programming,* vol. 89, pp. 149-185, 2000.
[113] E. Tan, "HYPER-WIDEBAND OFDM SYSTEM," MS, School of Electrical & Computer Engineering, Georgia Institute of Technology, 2016.
[114] D. Banerjee, S. Sen, S. K. Devarakond, and A. Chatterjee, "Power Aware Post-Manufacture Tuning of MIMO Receiver Systems," in *2012 25th International Conference on VLSI Design*, 2012, pp. 143-148.
[115] J. Han and G. Leus, "Space-Time and Space-Frequency Block Coded Vector OFDM Modulation," *IEEE Communications Letters,* vol. 21, pp. 204-207, 2017.
[116] S. Lukacs, A. V. Lutas, D. H. Lutas, and G. Sebestyen, "Hardware virtualization based security solution for embedded systems," in *Automation, Quality and Testing, Robotics, 2014 IEEE International Conference on*, 2014, pp. 1-6.
[117] *PUF – Physical Unclonable Functions Protecting next-generation Smart Card ICs with SRAM-based PUFs*. Available: http://www.nxp.com/documents/other/75017366.pdf
[118] Y. Jin, W. Xin, H. Sun, and Z. Chen, "PUF-Based RFID Authentication Protocol against Secret Key Leakage," in *Web Technologies and Applications*. vol. 7235, Q. Sheng, G. Wang, C. Jensen, and G. Xu, Eds., ed: Springer Berlin Heidelberg, 2012, pp. 318-329.
[119] C. Herder, M. D. Yu, F. Koushanfar, and S. Devadas, "Physical Unclonable Functions and Applications: A Tutorial," *Proceedings of the IEEE,* vol. 102, pp. 1126-1141, 2014.
[120] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical One-Way Functions," *Science,* vol. 297, pp. 2026-2030, September 20, 2002 2002.

[121] B. Gassend, D. Clarke, M. v. Dijk, and S. Devadas, "Silicon physical random functions," presented at the Proceedings of the 9th ACM conference on Computer and communications security, Washington, DC, USA, 2002.
[122] S. Devadas, E. Suh, S. Paral, R. Sowell, T. Ziola, and V. Khandelwal, "Design and Implementation of PUF-Based "Unclonable" RFID ICs for Anti-Counterfeiting and Security Applications," in *RFID, 2008 IEEE International Conference on*, 2008, pp. 58-64.
[123] D. Suzuki and K. Shimizu, "The Glitch PUF: A New Delay-PUF Architecture Exploiting Glitch Shapes," in *Cryptographic Hardware and Embedded Systems, CHES 2010*. vol. 6225, S. Mangard and F.-X. Standaert, Eds., ed: Springer Berlin Heidelberg, 2010, pp. 366-382.
[124] M. Kalyanaraman and M. Orshansky, "Novel strong PUF based on nonlinearity of MOSFET subthreshold operation," in *Hardware-Oriented Security and Trust (HOST), 2013 IEEE International Symposium on*, 2013, pp. 13-18.
[125] P. Tuyls, G.-J. Schrijen, B. Škorić, J. van Geloven, N. Verhaegh, and R. Wolters, "Read-Proof Hardware from Protective Coatings," in *Cryptographic Hardware and Embedded Systems - CHES 2006*. vol. 4249, L. Goubin and M. Matsui, Eds., ed: Springer Berlin Heidelberg, 2006, pp. 369-383.
[126] S. S. Kumar, J. Guajardo, R. Maes, G. J. Schrijen, and P. Tuyls, "Extended abstract: The butterfly PUF protecting IP on every FPGA," in *Hardware-Oriented Security and Trust, 2008. HOST 2008. IEEE International Workshop on*, 2008, pp. 67-70.
[127] R. Helinski, D. Acharyya, and J. Plusquellic, "A physical unclonable function defined using power distribution system equivalent resistance variations," in *Design Automation Conference, 2009. DAC '09. 46th ACM/IEEE*, 2009, pp. 676-681.
[128] M. Zaki and S. Tahar, "Formal verification of analog and mixed signal designs," in *Design and Test Workshop (IDT), 2009 4th International*, 2009, pp. 1-1.
[129] L. Yiming, H. Chih-Hong, L. Tien-Yeh, and H. Ming-Hung, "Process-Variation Effect, Metal-Gate Work-Function Fluctuation, and Random-Dopant Fluctuation in Emerging CMOS Technologies," *Electron Devices, IEEE Transactions on,* vol. 57, pp. 437-447, 2010.
[130] U. Ruhrmair, J. Solter, F. Sehnke, X. Xiaolin, A. Mahmoud, V. Stoyanova, *et al.*, "PUF Modeling Attacks on Simulated and Silicon Data," *Information Forensics and Security, IEEE Transactions on,* vol. 8, pp. 1876-1891, 2013.
[131] W. E. Cobb, E. D. Laspe, R. O. Baldwin, M. A. Temple, and Y. C. Kim, "Intrinsic Physical-Layer Authentication of Integrated Circuits," *Information Forensics and Security, IEEE Transactions on,* vol. 7, pp. 14-24, 2012.
[132] D. S. Board. (Feb 2005, "Task Force On HIGH PERFORMANCE MICROCHIP SUPPLY".
[133] P. Kumar and R. Srinivasan, "Detection of hardware Trojan in SEA using path delay," in *Electrical, Electronics and Computer Science (SCEECS), 2014 IEEE Students' Conference on*, 2014, pp. 1-6.
[134] H. Salmani, M. Tehranipoor, and J. Plusquellic, "A Novel Technique for Improving Hardware Trojan Detection and Reducing Trojan Activation Time," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on,* vol. 20, pp. 112-125, 2012.

[135] F. Wolff, C. Papachristou, S. Bhunia, and R. S. Chakraborty, "Towards Trojan-Free Trusted ICs: Problem Analysis and Detection Scheme," in *Design, Automation and Test in Europe, 2008. DATE '08*, 2008, pp. 1362-1365.

[136] J. Aarestad, D. Acharyya, R. Rad, and J. Plusquellic, "Detecting Trojans Through Leakage Current Analysis Using Multiple Supply Pad IDDQs," *Information Forensics and Security, IEEE Transactions on*, vol. 5, pp. 893-904, 2010.

[137] M. Banga and M. S. Hsiao, "A Novel Sustained Vector Technique for the Detection of Hardware Trojans," in *VLSI Design, 2009 22nd International Conference on*, 2009, pp. 327-332.

[138] C. Byeongju and S. K. Gupta, "Efficient Trojan Detection via Calibration of Process Variations," in *Test Symposium (ATS), 2012 IEEE 21st Asian*, 2012, pp. 355-361.

[139] B. Cha and S. K. Gupta, "Trojan detection via delay measurements: A new approach to select paths and vectors to maximize effectiveness and minimize cost," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2013*, 2013, pp. 1265-1270.

[140] D. Rai and J. Lach, "Performance of delay-based Trojan detection techniques under parameter variations," in *Hardware-Oriented Security and Trust, 2009. HOST '09. IEEE International Workshop on*, 2009, pp. 58-65.

[141] J. Yier and Y. Makris, "Hardware Trojan detection using path delay fingerprint," in *Hardware-Oriented Security and Trust, 2008. HOST 2008. IEEE International Workshop on*, 2008, pp. 51-57.

[142] X. Kan and M. Tehranipoor, "BISA: Built-in self-authentication for preventing hardware Trojan insertion," in *Hardware-Oriented Security and Trust (HOST), 2013 IEEE International Symposium on*, 2013, pp. 45-50.

[143] P. Song, F. Stellari, D. Pfeiffer, J. Culp, A. Weger, A. Bonnoit, *et al.*, "MARVEL : Malicious alteration recognition and verification by emission of light," in *Hardware-Oriented Security and Trust (HOST), 2011 IEEE International Symposium on*, 2011, pp. 117-121.

[144] S. Deyati, B. J. Muldrey, A. Singh, and A. Chatterjee, "High Resolution Pulse Propagation Driven Trojan Detection in Digital Logic: Optimization Algorithms and Infrastructure," in *Test Symposium (ATS), 2014 IEEE 23rd Asian*, 2014, pp. 200-205.

[145] T. Hoque, M. Mustapa, F. Amsaad, and M. Niamat, "Assessment of NAND based ring oscillator for hardware Trojan detection," in *Circuits and Systems (MWSCAS), 2015 IEEE 58th International Midwest Symposium on*, 2015, pp. 1-4.

[146] C. Yuan, C. Chip-Hong, and C. Shoushun, "A Cluster-Based Distributed Active Current Sensing Circuit for Hardware Trojan Detection," *Information Forensics and Security, IEEE Transactions on*, vol. 9, pp. 2220-2231, 2014.

[147] S. Narasimhan, Y. Wen, W. Xinmu, S. Mukhopadhyay, and S. Bhunia, "Improving IC Security Against Trojan Attacks Through Integration of Security Monitors," *Design & Test of Computers, IEEE*, vol. 29, pp. 37-46, 2012.

[148] R. S. Chakraborty, S. Narasimhan, and S. Bhunia, "Hardware Trojan: Threats and emerging solutions," in *High Level Design Validation and Test Workshop, 2009. HLDVT 2009. IEEE International*, 2009, pp. 166-171.

[149] S. Narasimhan, D. Dongdong, R. S. Chakraborty, S. Paul, F. Wolff, C. Papachristou, *et al.*, "Multiple-parameter side-channel analysis: A non-invasive hardware Trojan detection approach," in *Hardware-Oriented Security and Trust (HOST), 2010 IEEE International Symposium on*, 2010, pp. 13-18.
[150] "http:/ptm.asu.edu."
[151] G. T. Becker, F. Regazzoni, C. Paar, and W. P. Burleson, "Stealthy Dopant-Level Hardware Trojans," in *Cryptographic Hardware and Embedded Systems - CHES 2013: 15th International Workshop, Santa Barbara, CA, USA, August 20-23, 2013. Proceedings*, G. Bertoni and J.-S. Coron, Eds., ed Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 197-214.
[152] A. I. Kayssi, K. A. Sakallah, and T. M. Burks, "Analytical transient response of CMOS inverters," *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on*, vol. 39, pp. 42-45, 1992.
[153] L. Bisdounis, S. Nikolaidis, and O. Koufopavlou, "Analytical transient response and propagation delay evaluation of the CMOS inverter for short-channel devices," *Solid-State Circuits, IEEE Journal of*, vol. 33, pp. 302-306, 1998.
[154] L. W. Massengill and P. W. Tuinenga, "Single-Event Transient Pulse Propagation in Digital CMOS," *Nuclear Science, IEEE Transactions on*, vol. 55, pp. 2861-2871, 2008.
[155] X. Gili, S. Barcelo, S. A. Bota, and J. Segura, "Analytical Modeling of Single Event Transients Propagation in Combinational Logic Gates," *Nuclear Science, IEEE Transactions on*, vol. 59, pp. 971-979, 2012.
[156] J. P. Roth, "Diagnosis of Automata Failures: A Calculus and a Method," *IBM Journal of Research and Development*, vol. 10, pp. 278-291, 1966.
[157] P. Goel and B. C. Rosales, "PODEM-X: An Automatic Test Generation System for VLSI Logic Structures," in *Design Automation, 1981. 18th Conference on*, 1981, pp. 260-268.
[158] I. Sutherland, B. Sproull, and D. Harris, *Logical effort: designing fast CMOS circuits*: Morgan Kaufmann Publishers Inc., 1999.